Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

“САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ”

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

РАЗРАБОТКА ЭФФЕКТИВНЫХ МЕТОДОВ УЧЕТА ПРИМЕРОВ ПОВЕДЕНИЯ ПРИ СИНТЕЗЕ АВТОМАТНЫХ МОДЕЛЕЙ ПО ТЕМПОРАЛЬНЫМ ФОРМУЛАМ

Автор *Давлетшин Руслан Олегович* _____

Направление подготовки *01.03.02 Прикладная математика и информатика*

Квалификация *Бакалавр*

Руководитель *Ульянцев В. И., к.т.н* _____

К защите допустить

Зав. кафедрой КТ *Васильев В.Н., проф., д.т.н.* _____

“___” _____ 2018 г.

Санкт-Петербург, 2018 г.

Студент *Давлетшин Р. О.* Группа *М3436* Кафедра *КТ* Факультет *ИТиП*

Направленность (профиль), специализация
Математические модели и алгоритмы в разработке программного обеспечения

ВКР принята «____» _____ 2018 г.

Оригинальность ВКР _____%

ВКР выполнена с оценкой _____

Дата защиты «____» июня 2018 г.

Секретарь ГЭК *Павлова О.Н.* _____

Листов хранения _____

Демонстрационных материалов/Чертежей хранения *отсутствуют*

Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

“САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ”

УТВЕРЖДАЮ

Зав. кафедрой *КТ*

проф. Васильев В. Н

_____ 2018 г.

З А Д А Н И Е
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

Студенту *Давлетшин Р. О.* **Группа** *М3436* **Кафедра** *КТ* **Факультет** *ИТиП*

Руководитель *Ульянцев В.И., к.т.н., доцент кафедры КТ*

1 Наименование темы: *Разработка эффективных методов учета примеров поведения при синтезе автоматных моделей по темпоральным формулам*

Направление подготовки (специальность) *01.03.02 Прикладная математика и информатика*

Квалификация *Бакалавр*

2. Срок сдачи студентом законченной работы *15 мая 2018 г.*

3. Техническое задание и исходные данные к диссертации:

Требуется разработать методы эффективного учета примеров поведения при синтезе автоматных моделей по темпоральным формулам с помощью сведения к задаче выполнимости булевой формулы

4 Содержание выпускной работы (перечень подлежащих разработке вопросов):

1. *Обзор предметной области, обозначение роли примеров поведения, постановка задачи, анализ существующих подходов.*

2. *Описание методов эффективного учета примеров поведения при синтезе автоматных моделей по темпоральным формулам с помощью сведения к задаче выполнимости булевой формулы.*

3. *Проведение экспериментального исследования разработанных методов*

5 Перечень графического материала (с указанием обязательного материала)

Графические материалы и чертежи работой не предусмотрены

6 Исходные материалы и пособия

1. *J. Hopcroft, R. Motwani, J. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, 2006.*

2. *Peter Faymonville, Bernd Finkbeiner, Markus N. Rabe, Leander Tentrup: Encodings of Bounded Synthesis, 2017.*

7 Дата выдачи задания «21» «ноября» 2017г.

Руководитель _____

Задание принял к исполнению_____

«21» «ноября» 2017г.

Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

“САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ”

АННОТАЦИЯ

ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

Студента *Давлетшин Руслан Олегович*

Наименование темы ВКР: *Разработка эффективных методов учета примеров поведения при синтезе автоматных моделей по темпоральным формулам*

Наименование организации, где выполнена ВКР *Университет ИТМО*

ХАРАКТЕРИСТИКА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

1 Цель исследования: *Разработать эффективные методы учета примеров поведения при синтезе автоматных моделей по темпоральным формулам с помощью сведения к задаче удовлетворения булевой формулы*

2 Задачи, решаемые в ВКР:

1) *Обзор существующих решений учета примеров поведения*
2) *Разработка методов учета примеров поведения при синтезе автоматных моделей по темпоральным формулам с помощью сведения к задаче выполнимости булевой формулы без кванторов*
3) *Разработка методов учета примеров поведения при синтезе автоматных моделей по темпоральным формулам с помощью сведения к задаче выполнимости булевой формулы с кванторами*
4) *Проведение экспериментального исследования разработанных методов*

3 Число источников, использованных при составлении обзора: *10*

4 Полное число источников, использованных в работе: *12*

5 В том числе источников по годам

Отечественных			**Иностранных**		
Последние 5 лет	От 5 до 10 лет	Более 10 лет	Последние 5 лет	От 5 до 10 лет	Более 10 лет
2	1	0	4	2	3

6 Использование информационных ресурсов Internet: *Да, 2*

7 Использование современных пакетов компьютерных программ и технологий

Пакеты компьютерных программ и технологий	**Параграф работы**
Язык программирования Swift версии 4.0.3	3
Среда разработки AppCode версии 2018.1.1	3
Среда разработки Xcode 9.2	3
Язык программирования Python 2.7	3
Среда разработки PyCharm Professional версии 2018.1.1	3

8 Краткая характеристика полученных результатов:
В результате выполнения работы были получены новые методы учета примеров поведения при синтезе автоматных моделей по темпоральным формулам, продемонстрировавшие результаты значительно превосходящие базовые решения

9 Полученные гранты, при выполнении работы:
При выполнении работы грантов получено не было.

10 Наличие публикаций и выступлений на конференциях по теме выпускной работы *нет*

Студент *Давлетшин Р. О.* _____

Руководитель *Ульянцев В. И.* _____

“_____” _____ 2018 г.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Синтез автоматных моделей — это довольно распространенная задача. Область её применения варьируется от проверки программного обеспечения и синтеза управляющих систем [1] и [2], до задач биоинформатики. Распространённым способом решения данной задачи является сведение к задаче о выполнимости булевой формулы [3].

Обычно для задания спецификации синтезируемой системы используется формулы линейной темпоральной логики. В данной области недавно появился новый перспективный подход синтеза автоматной модели с ограничением на размер итоговой системы [2]. Его авторы два года подряд занимают призовые места в соревновании по синтезу автоматных систем [4], опережая конкурентов как в скорости, так и в качестве синтезируемых реактивных систем.

Однако зачастую только формул темпоральной логики недостаточно, чтобы задать все особенности синтезируемой системы. В таких случаях используются примеры поведения [1] и [5]. Также иногда задача состоит в синтезе автоматных систем только по примерам поведения. К сожалению подход, описанный в [2] поддерживает представление примеров поведения только в виде темпоральных формул, что снижает производительность данного подхода.

Целью данной работы является разработка новых эффективных методов учета примеров поведения, основанных на подходе, представленном в [2], и сравнение новых методов с другими подходами к представлению примеров поведения.

ГЛАВА 1. ОБЗОР

В данной главе даны определения основным терминам и понятиям, используемым в работе. Также в настоящей главе описывается задача синтеза автоматных систем, обозначается роль примеров поведения, описывается перспективный алгоритм синтеза автоматных моделей через сведение к задаче выполнимости булевой формулы, который не поддерживает использование примеров поведения, заданных способом, отличным от темпоральных формул, и проводятся анализ других подходов, поддерживающими учет примеров поведения в спецификации наряду с темпоральными формулами.

1.1. Термины и понятия

В данном разделе даны определения основным терминам и понятиям, которые используются в работе.

1.1.1. Задача выполнимости булевой формулы

Задача выполнимости булевой формулы (boolean satisfiability problem, SAT) — задача о поиске такого назначения пропозициональных переменных, входящих в булеву формулу, чтобы данная формула стала истинной. Для задачи SAT формула может содержать только переменные, скобки и операции И, ИЛИ и НЕ. В задаче SAT для связи входящих в формулу пропозициональных переменных подразумевается использование квантора существования. Задача выполнимости булевой формулы с кванторами (quantified boolean satisfiability problem, QSAT) является расширением задачи SAT, в котором наряду с квантором существования для связи переменных может использоваться квантор всеобщности.

1.1.2. Синтез с ограничением на размер системы

Синтез с ограничением на размер системы (bounded synthesis) — подход к синтезу автоматных моделей с ограничением на размер итоговой системы и на число посещений отвергающих состояний. Задача синтеза с ограничением на размер системы может быть представлена как задача о разрешимости системы

ограничений, даже в таких условиях, где другие подходы к синтезу неразрешимы, например, при синтезе асинхронных или распределенных систем [6].

1.1.3. Линейная темпоральная логика

Линейная темпоральная логика (Linear temporal logic, LTL) — это логика с операторами, позволяющими работать со временем. С её помощью можно задавать порядок явлений и их взаимодействия во времени. Помимо обычных логических операторов (отрицание, конъюнкция, дизъюнкция, импликация, логическое равенство) формулы линейной темпоральной логики поддерживает следующие операторы:

- $\mathbf{X}\varphi$ (next) — формула φ должна выполняться в следующем состоянии.
- $\mathbf{F}\varphi$ (finally) — формула φ должна выполниться в одном из следующих состояний.
- $\mathbf{G}\varphi$ (globally) — формула φ должна выполняться в каждом состоянии.
- $\varphi\mathbf{U}\psi$ (until) — формула φ должна выполняться хотя бы до того момента, как выполнится формула ψ (которая обязательно должна выполнится сейчас или в будущем).
- $\varphi\mathbf{R}\psi$ (release) — формула ψ должна выполняться до того состояния (включая текущее состояние), в котором φ впервые выполнится, если φ никогда не выполнится, то ψ должна выполняться всегда.
- $\varphi\mathbf{W}\psi$ (weak until) — формула φ должна выполняться хотя бы до того момента, как выполнится формула ψ, если ψ никогда не выполнится, то φ должна выполняться всегда.
- $\varphi\mathbf{M}\psi$ (strong release) — формула ψ должна выполняться до того состояния (включая это состояние), как φ впервые выполнится (которая обязательно должна выполнится сейчас или в будущем).

Пусть Σ — конечный набор пропозициональных переменных и линейная темпоральная формула φ задана над этим набором, тогда язык формулы φ, обозначаемый как $\mathcal{L}(\varphi)$, состоит из бесконечных последовательностей состояний $\sigma \in (2^{\Sigma})^*$.

Пример формулы линейной темпоральной логики:

$$G(\neg g_0 \vee \neg g_1) \wedge G(r_0 \rightarrow F g_0) \wedge G(r_1 \rightarrow F g_1)$$

Данная формула задаёт поведение системы арбитра, которая после запроса r_0 или r_1 рано или поздно должна выдать соответствующее разрешение, при этом запрещается выдавать оба разрешения одновременно.

1.1.4. Универсальный автомат ко-Бюхи

Универсальный автомат ко-Бюхи (universal co-Büchi automaton) $\mathcal{A}$ над конечным алфавитом Σ задается четверкой $\langle Q, q_0, \delta, F \rangle$, где Q — конечный набор состояний автомата, $q_0 \in Q$ — начальное состояние автомата, δ: $Q \times 2^{\Sigma} \times Q$ — отношение перехода и $F \subseteq Q$ — набор отвергающих состояний. Пусть дано бесконечное слово $\sigma \in (2^{\Sigma})^*$, запуск заданного слова на автомате $\mathcal{A}$ порождает бесконечную последовательность состояний $q_0 q_1 q_2 \ldots \in Q^*$. Запуск считается допускающим, если содержит лишь конечное число отвергающих состояний. Автомат $\mathcal{A}$ принимает слово σ, если все запуски заданного слова на автомате $\mathcal{A}$ были принимающими. Язык автомата $\mathcal{A}$ обозначается как $\mathcal{L}(\mathcal{A})$ и представляет собой множество $\{\sigma \in (2^{\Sigma})^* \mid \mathcal{A} \text{ принимает } \sigma\}$.

Далее универсальные автоматы ко-Бюхи будут изображаться в виде ориентированных графов с вершинами Q и символьным представлением отношения δ в виде пропозициональных булевых формул $\mathbb{B}(\Sigma)$. Отвергающие состояния F будут отмечены двойными линиями. Пример автомата ко-Бюхи, соответствующего приведенной выше LTL формуле, задающей поведение системы арбитра, можно увидеть на рис. 1.

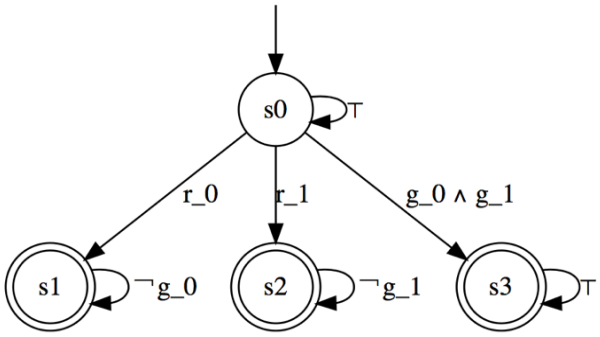

Рисунок 1 — Пример автомата ко-Бюхи $\mathcal{A}$

1.1.5. Система переходов

Пусть Σ — набор пропозициональных переменных, разделим его на две части I — переменные, контролируемые средой и O — переменные, контролируемые системой. В данном определении под средой понимается внешняя среда, с которой взаимодействует автоматная система, а под термином система непосредственно сама автоматная система. Таким образом предполагается, что переменные из множества I будут изменяться только внешней средой, а переменные из множества O — только автоматной системой. Система переходов (transition system) $\mathcal{T}$ задается тройкой $\langle T, t_0, \tau \rangle$, где T — конечный набор состояний, t_0 — начальное состояние и $\tau: T \times 2^I \to 2^O \times T$ — функция переходов. Если для данного состояния $t \in T$ и переменных $\boldsymbol{i} \neq \boldsymbol{i}' \in 2^I$ из $\tau(t, \boldsymbol{i}) = (\boldsymbol{o}, \_)$ и $\tau(t, \boldsymbol{i}') = (\boldsymbol{o}', \_)$ следует, что $\boldsymbol{o} = \boldsymbol{o}'$, то система переходов называется системой переходов Мура, иначе — системой переходов Мили.

Запуск бесконечной последовательности $\boldsymbol{i_0 i_1} \ldots \in (2^I)^*$ на $\mathcal{T}$ порождает бесконечный след $(\{t_0\} \cup \boldsymbol{i_0} \cup \boldsymbol{o_0})(\{t_1\} \cup \boldsymbol{i_1} \cup \boldsymbol{o_1}) \ldots \in (2^{T \cup I \cup O})^*$, где $\tau(t_j, i_j) = (o_j, t_{j+1})$ для каждого $j \geq 0$. Путь $\omega \in (2^{I \cup O})^*$ — это проекция следа на пропозициональные переменные. Множество всех путей, порожденных системой переходов $\mathcal{T}$, обозначим как $Paths(\mathcal{T})$. Система переходов реализует формулу линейной темпоральной логики, если $Paths(\mathcal{T}) \subseteq \mathcal{L}(\varphi)$. Пример минимальной системы переходов, реализующей приведенную ранее формулу темпоральной логики, которая представляет систему арбитра, изображен на рис. 2.

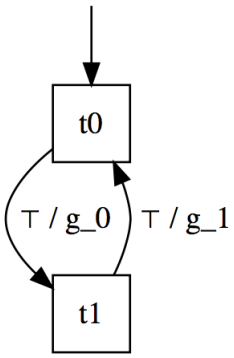

Рисунок 2 — Пример системы переходов $\boldsymbol{\mathcal{T}}$

1.1.6. Граф обхода

Произведением системы переходов $\mathcal{T} = \langle T, t_0, \tau \rangle$ и автомата ко-Бюхи $\mathcal{A} = \langle \mathrm{Q}, q_0, \delta, \mathrm{F} \rangle$ является граф обхода $\mathcal{G} = \langle V, E \rangle$, где $V = T \times Q$ — множество вершин и $E \subseteq V \times V$ — множество ребер такое, что $\big((t,q),(t',q')\big) \in E \Leftrightarrow \exists \boldsymbol{i} \in 2^I . \exists \boldsymbol{o} \in 2^O . \tau(t,\boldsymbol{i}) = (\boldsymbol{o}, t')$ и $(q, \boldsymbol{i} \cup \boldsymbol{o}, q') \in \delta$.

Пример графа обхода, полученного как произведение автомата ко-Бюхи с рис. 1 и системы переходов с рис. 2, представлен на рис. 3.

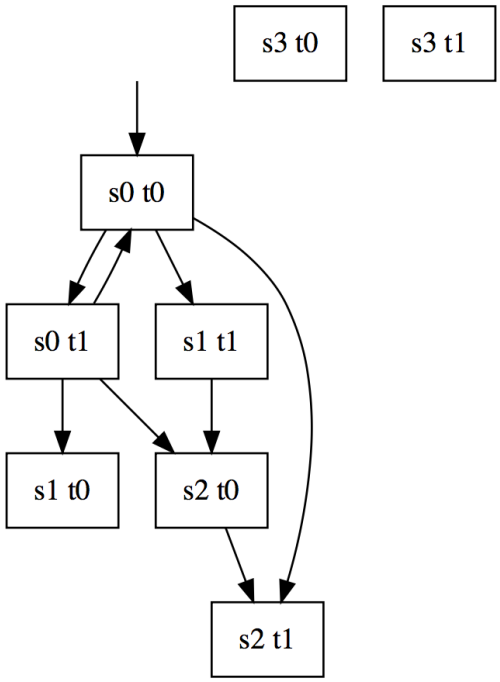

Рисунок 3 — Граф обхода для автомата ко-Бюхи $\mathcal{A}$ и системы переходов $\boldsymbol{\mathcal{T}}$

1.2. Задача синтеза автоматных систем

Задача синтеза реактивных автоматных систем состоит в поиске автоматной системы минимального размера, которая будет реализовывать заданную спецификацию. Существует множество подходов к решению данной задачи, но одним из самых успешных считается сведение к задаче о выполнимости булевой формулы [3]. Основным плюсом такого подхода кроме скорости, за которую находится целевая система, является тот факт, что при улучшении программ для решения задачи о выполнимости булевой формулы, будет также улучшаться работа алгоритма поиска целевой реактивной системы.

1.3. Примеры поведения

Обычно спецификация для целевой реактивной системы задается в виде формул линейной темпоральной логики, однако зачастую есть необходимость дополнительно указать примеры поведения искомой системы. В рамках данной работы будем называть сценариями последовательности пар векторов $\boldsymbol{i} \in 2^{|I|}$ и $\boldsymbol{o} \in 2^{|O|}$, задающие состояние переменных, контролируемых средой, и переменных, контролируемых системой. Примерами поведения будем называть набор из нескольких сценариев. Отдельную пару, из которых состоит сценарий, будем называть элементом сценария. Считается, что реактивная система реализует примеры поведения, если в ней можно воспроизвести каждый сценарий из данного набора.

Примеры поведения позволяют указать дополнительные свойства системы, которые не были представлены в LTL спецификации. Также зачастую ставится задача синтеза реактивной системы только по примерам поведения [7].

1.4. Bounded synthesis

В настоящем разделе описывается перспективный подход к синтезу автоматных систем, предложенный в [2]. Его разработчики уже на протяжении двух лет удерживают лидерство в соревновании по синтезу автоматных систем SYNTCOMP 2016 и SYNTCOMP 2017 [4]. Однако у данного подхода есть свои недостатки, одним из которых является отсутствие возможности учета

примеров поведения, за исключением случаев, когда сценарии представлены в виде линейных темпоральных формул, что не очень эффективно. Задачей данной работы является разработка эффективных методов учета примеров поведения при использовании подхода bounded synthesis. Новые методы позволят одновременно учесть в одной SAT или QSAT формуле как LTL спецификацию, так и примеры поведения, в отличие от других методов [5], что должно позволить быстрее находить автоматную систему минимального размера.

Синтез с ограничением на размер системы [6] — это процесс синтеза системы переходов оптимального размера по спецификации, заданной в виде формулы линейной темпоральной логики. По заданной формуле φ строится автомат ко-Бюхи $\mathcal{A}$, который принимает язык $\mathcal{L}(\varphi)$. Система переходов $\mathcal{T}$ реализует спецификацию, заданную формулой φ, тогда и только тогда, когда все пути, порожденные системой переходов, лежат в языке $\mathcal{L}(\varphi)$. Система переходов $\mathcal{T}$ принимается автоматом ко-Бюхи $\mathcal{A}$, если в графе запусков, который является произведением $\mathcal{T}$ и $\mathcal{A}$, все пути содержат лишь конечное количество посещений отвергающих состояний.

Подход, применяемый при синтезе с ограничением на размер системы, заключается в разрешении системы ограничений, доказывающей существование системы переходов $\mathcal{T}$ вместе с корректной функцией аннотации.

1.4.1. Функция аннотации

Функция аннотации $\lambda: T \times Q \to \{\bot\} \cup \mathbb{N}$ — это функция, которая присваивает вершинам графа запусков либо натуральное число k, либо $\bot$ в случае, если вершина недостижима. Функция аннотации считается корректной, если выполнены следующие условия:

- паре начальных состояний (t_0, q_0) присвоено натуральное число,
- если паре состояний (t, q) присвоено натуральное число k, то для всех $\boldsymbol{i} \in 2^I$ и $\boldsymbol{o} \in 2^O$ таких, что $\tau(t, \boldsymbol{i}) = (\boldsymbol{o}, t')$ и $(q, \boldsymbol{i} \cup \boldsymbol{o}, q') \in \delta$, паре (t', q') должно быть присвоено большее или равное

натуральное число, или строго большее в случае, если $q' \in F$. То есть $\lambda(t', q') \rhd_{q'} k$, где $\rhd_{q'} := >$, если $q' \in F$, $\geq$ иначе.

1.4.2. Система ограничений

В данном подразделе описывается построение системы ограничений по заданным $\mathcal{T}$, $\mathcal{A}$ и λ, которая разрешима только в том случае, когда функция аннотации корректна. По определению, корректность функции аннотации можно доказать через проверку всех переходов в графе, для этого закодируем $\mathcal{T}$, $\mathcal{A}$ и λ следующими пропозициональными переменными:

- Функцию переходов τ системы переходов $\mathcal{T}$ представим двумя пропозициональными переменными: $o_{t,\boldsymbol{i}}$ для каждой исходящей переменной, контролируемой системой, $o \in O$ и $\tau_{t,\boldsymbol{i},t'}$, обозначающей переход из состояния t в состояние t'. Пусть $(t, t') \in T \times T$ и $\boldsymbol{i} \in 2^I$, тогда $\tau_{t,\boldsymbol{i},t'} = True \Leftrightarrow \tau(t, \boldsymbol{i}) = (\_, t')$ и $o_{t,\boldsymbol{i}} = True \Leftrightarrow \tau(t, \boldsymbol{i}) = (\boldsymbol{o}, \_)$, где $o \in \boldsymbol{o}$.
- Отношения перехода $\delta: Q \times 2^{I \cup O} \times Q$ автомата ко-Бюхи $\mathcal{A}$ зададим в виде формулы $\delta_{t,q,\boldsymbol{i},q'}$ над переменными $o_{t,\boldsymbol{i}}$ таким образом, что назначение $\boldsymbol{o}$ для переменных $o_{t,\boldsymbol{i}}$ удовлетворяет $\delta_{t,q,\boldsymbol{i},q'} \Leftrightarrow (q, \boldsymbol{i} \cup \boldsymbol{o}, q') \in \delta$.
- Для удобства разделим функцию аннотации на две части: $\lambda^{\mathbb{B}}: T \times Q \to \mathbb{B}$, отображающую достижимость вершины, и $\lambda^{\#}: T \times Q \to \mathbb{N}$. Для каждых $t \in T$ и $q \in Q$ введем переменную $\lambda^{\mathbb{B}}_{t,q} = True \Leftrightarrow$ вершина (q, t) в графе запусков достижима из начальной вершины, и переменную $\lambda^{\#}_{t,q}$, представленную битовым вектором и равную значению $\lambda(t, q)$.

Используя описанные выше пропозициональные переменные, составляется булева формула, проверяющая корректность функции аннотации:

$$\bigwedge_{q \in Q} \bigwedge_{t \in T} \left(\lambda^{\mathbb{B}}_{t,q} \to \bigwedge_{q' \in Q} \bigwedge_{\boldsymbol{i} \in 2^I} \left(\delta_{t,q,\boldsymbol{i},q'} \to \bigwedge_{t' \in T} \left(\tau_{t,\boldsymbol{i},t'} \to \lambda^{\mathbb{B}}_{t',q'} \wedge \lambda^{\#}_{t',q'} \rhd_{q'} \lambda^{\#}_{t,q} \right) \right) \right)$$

Если для данных $\mathcal{T}$, $\mathcal{A}$ и λ пропозициональные переменные удовлетворяют систему ограничений, то функция аннотации корректна [8].

1.4.3. Сведение к задаче SAT

Рассмотрим задачу поиска системы переходов с корректной аннотацией. Для этого возьмём формулу, описанную в предыдущем подразделе, которую вместо проверки на корректность заданного назначения пропозициональных переменных для $\mathcal{T}$, $\mathcal{A}$ и λ будем использовать для поиска удовлетворяющего назначения. Сама формула остается почти без изменений, необходимо лишь добавить два предиката: предикат, гарантирующий корректность переменных $\lambda^{\mathbb{B}}_{t,q}$, отвечающих за достижимость вершин в графе запусков, и предикат, отвечающий за то, что функция перехода, заданная переменными $\tau_{t,\boldsymbol{i},t'}$, для каждого состояния имеет переходы для всех возможных вариантов переменных, контролируемых средой. Для первого предиката достаточно включить в формулу переменную $\lambda^{\mathbb{B}}_{t_0,q_0}$, так как всё остальное в формуле уже учтено. Итоговая формула выглядит следующим образом:

$$\exists\left\{\lambda^{\mathbb{B}}_{t,q}, \lambda^{\#}_{t,q} \middle| t \in T, q \in Q\right\}$$

$$\exists\left\{\tau_{t,\boldsymbol{i},t'} \middle| (t,t') \in T\times T, \boldsymbol{i} \in 2^I\right\}$$

$$\exists\left\{o_{t,\boldsymbol{i}} \middle| o \in O, t \in T, \boldsymbol{i} \in 2^I\right\}$$

$$\lambda^{\mathbb{B}}_{t_0,q_0} \wedge \left(\bigwedge_{t\in T}\bigwedge_{\boldsymbol{i}\in 2^I}\bigvee_{t'\in T}\tau_{t,\boldsymbol{i},t'}\right) \wedge$$

$$\bigwedge_{q\in Q}\bigwedge_{t\in T}\left(\lambda^{\mathbb{B}}_{t,q} \to \bigwedge_{q'\in Q}\bigwedge_{\boldsymbol{i}\in 2^I}\left(\delta_{t,q,\boldsymbol{i},q'} \to \bigwedge_{t'\in T}\left(\tau_{t,\boldsymbol{i},t'} \to \lambda^{\mathbb{B}}_{t',q'} \wedge \lambda^{\#}_{t',q'} \rhd_{q'} \lambda^{\#}_{t,q}\right)\right)\right)$$

Формула имеет размер $\mathcal{O}(nm^2 \cdot 2^{|I|} \cdot (|\delta_{q,q'}| + n\log(nm)))$ дизъюнктов и содержит $\mathcal{O}(n(m\log(nm) + 2^{|I|} \cdot (|O| + n)))$ переменных, где $n = |T|$ и $m = |Q|$ [2].

Так как в формуле используются только кванторы существования, она может быть решена с помощью программного средства для решения задачи

SAT. Далее по найденному удовлетворяющему назначению пропозициональных переменных строится искомая система переходов.

1.4.4. Сведение к задаче QSAT

Основным недостатком предложенной выше формулы является явный перебор всех переменных среды $\boldsymbol{i} \in 2^I$ при работе с пропозициональными переменными $\tau_{t,\boldsymbol{i},t'}$ и $o_{t,\boldsymbol{i}}$, задающими функцию перехода. Из-за этого размер формулы и количество переменных в ней экспоненциально зависит от количества переменных среды. Также из-за того, что все кванторы существования находятся на одном уровне, теряется некоторая часть информации, специфичной для данной задачи, и программное средство для решения задачи SAT вынуждено искать значения для всех переменных, что может приводить к не минимальным решениям τ и o из-за неявных внутренних зависимостей.

Проблемы, описанные выше, решаются с помощью добавления квантора всеобщности по переменным среды. После чего мы получаем булеву формулу с кванторами и избавляемся от экспоненциальной зависимости. Все пропозициональные переменные, стоящие под кванторами существования, разделяются на две части — внешнюю, независимую от переменных среды, и внутреннюю. Так как пропозициональные переменные, задающие функцию аннотации λ, не должны зависеть от переменных среды, то они находятся во внешней части, остальные пропозициональные переменные находятся во внутренней. Таким образом получается, что переменные, представляющие функцию перехода, могут восприниматься как булевы функции над множеством I. В связи с этим индексы $\boldsymbol{i}$ в переменных $\tau_{t,\boldsymbol{i},t'}$ и $o_{t,\boldsymbol{i}}$ можно опустить. Также теперь отношение перехода $\delta: Q \times 2^{I \cup O} \times Q$ задается в виде пропозициональной формулы над переменными среды I и пропозициональными переменными o_t, таким образом, что назначение $\boldsymbol{i} \cup \boldsymbol{o}$ удовлетворяет формулу $\delta_{t,q,q'} \Leftrightarrow (q, \boldsymbol{i} \cup \boldsymbol{o}, q') \in \delta$. Модифицированная формула имеет следующий вид:

$$\exists\left\{\lambda_{t,q}^{\mathbb{B}}, \lambda_{t,q}^{\#} \middle| t \in T, q \in Q\right\}$$

$$\forall I$$

$$\exists\left\{\tau_{t,t'} \middle| (t,t') \in T \times T\right\}$$

$$\exists\{o_t | o \in O, t \in T\}$$

$$\lambda_{t_0,q_0}^{\mathbb{B}} \wedge \left(\bigwedge_{t\in T} \bigvee_{t'\in T} \tau_{t,t'}\right) \wedge$$

$$\bigwedge_{q\in Q} \bigwedge_{t\in T} \left(\lambda_{t,q}^{\mathbb{B}} \rightarrow \bigwedge_{q'\in Q} \left(\delta_{t,q,q'} \rightarrow \bigwedge_{t'\in T} \left(\tau_{t,t'} \rightarrow \lambda_{t',q'}^{\mathbb{B}} \wedge \lambda_{t',q'}^{\#} \rhd_{q'} \lambda_{t,q}^{\#}\right)\right)\right)$$

Размер новой формулы составляет $\mathcal{O}(nm^2 \cdot (|\delta_{q,q'}| + n\log(nm)))$ дизъюнктов, а количество переменных, находящихся под кванторами существования и всеобщности, равно $\mathcal{O}(n(m\log(nm) + |O| + n))$ и $\mathcal{O}(|I|)$ соответственно [2].

1.5. Альтернативные подходы к синтезу автоматных систем с учетом примеров поведения

Существует несколько различных подходов к синтезу автоматных систем, которые позволяют учитывать одновременно спецификацию в виде формул линейной темпоральной логики и примеры поведения. Среди них можно выделить следующие методы:

- Итеративный подход на основе сведения к задаче SAT [5], работа которого состоит из нескольких этапов: представлении сценариев в виде дерева сценариев, которое затем представляется как булева формула, генерации по полученной формуле автоматной системы, проверки синтезированной автоматной системы с помощью подхода проверки моделей [9] на соответствие линейным темпоральным формулам, если на текущем этапе выявляются контрпримеры, то они добавляются в дерево сценариев со специальной пометкой и весь процесс повторяется заново. Таким

образом постепенно выводится автомат, реализующий заданную спецификацию, или выявляется факт, отрицающий существование такой системы. Важным пунктом в описанном подходе является использование программных средств для решения задачи SAT, поддерживающих инкрементальный поиск решения [10], которые позволяют не перезапускать полностью заново поиск решения после добавления очередного контрпримера.

- Другим подходом является метод [1], основывающийся на сведении к задаче QSAT и проверке моделей с ограничением на длину проверяемых путей [9]. Настоящий подход чем-то похож на предыдущий, однако теперь вместо добавления контрпримеров последовательно увеличивается граница, используемая при проверке моделей, которая отвечает за длину проверяемых путей. После генерации автомата, как и предыдущем методе, снова выполнятся проверка удовлетворяет ли полученный автомат LTL свойствам, если нет, то увеличивается граница, используемая при проверке моделей, и операция повторяется заново. Также существует модификация данного подхода [1], в котором QSAT формула переводится в SAT формулу путем почти экспоненциального раскрытия кванторов всеобщности.

Выводы по главе 1

В данной главе были даны определения терминам и понятиям, используемым в работе, была произведена постановка задачи синтеза автоматных систем по спецификации в виде темпоральных формул и примеров поведения, описан перспективный алгоритм сведения задачи синтеза автоматных систем к задаче о выполнимости булевой формулы, на котором будет основываться данная работа, а также проведен анализ предметной области и других подходов.

ГЛАВА 2. ТЕОРЕТИЧЕСКОЕ ИССЛЕДОВАНИЕ

В данной главе будут приведены разработанные методы учета примеров поведения при сведении к задаче о выполнимости булевой формулы и проведен аналитический анализ их эффективности. Также в настоящей главе будут описаны способы построения деревьев и графов сценариев с кластеризацией и без, необходимых для эффективного кодирования сценариев при сведении к задаче о выполнимости булевой формулы.

2.1. Представление примеров поведения в виде LTL формул

В текущий момент единственным способом представления примеров поведения, предполагающим возможность их добавления в спецификацию при использовании подхода синтеза с ограничением на размер системы, является представление примеров поведения в виде формул линейной темпоральной логики. Так как данный метод не требует модификаций алгоритма сведения к задаче выполнимости булевой формулы, используемого при подходе bounded synthesis, будем считать его точкой отправления при дальнейших сравнениях результатов с другими предложенными методами.

Для представления примеров поведения в виде линейных темпоральных формул необходимо последовательно заменить каждый переход в списке элементов конкретного сценария на конструкцию вида

$$\boldsymbol{i}_k \rightarrow \boldsymbol{o}_k \wedge \mathrm{X}(\boldsymbol{i}_{k+1} \rightarrow \boldsymbol{o}_{k+1} \wedge \dots)$$

где вектор $\boldsymbol{i}$ задаёт состояние переменных среды в данном элементе сценария, вектор $\boldsymbol{o}$ задаёт состояние переменных системы, а X означает оператор темпоральной логики next.

Допустим, что $I = \{e_{11}, e_{12}, e_2, e_3, e_4\}$, $O = \{z_1, z_2, z_3\}$ и имеются следующие сценарии:

$(e_{11}|z_1) \rightarrow (e_2|\,) \rightarrow (e_{12}|z_2) \rightarrow (e_2|\,)$

$(e_{11}|z_1) \rightarrow (e_4|z_3)$

Тогда темпоральные формулы, задающие эти сценарии, будут выглядеть следующим образом:

$$(e_{11} \wedge \neg e_4 \wedge \neg e_{12} \wedge \neg e_3 \wedge \neg e_2)$$
$$\rightarrow (z_1 \wedge \neg z_2 \wedge \neg z_3 \wedge X((e_2 \wedge \neg e_{11} \wedge \neg e_4 \wedge \neg e_{12} \wedge \neg e_3)$$
$$\rightarrow (\neg z_1 \wedge \neg z_2 \wedge \neg z_3 \wedge X((e_{12} \wedge \neg e_{11} \wedge \neg e_4 \wedge \neg e_3 \wedge \neg e_2)$$
$$\rightarrow (z_2 \wedge \neg z_1 \wedge \neg z_3 \wedge X((e_2 \wedge \neg e_{11} \wedge \neg e_4 \wedge \neg e_{12} \wedge \neg e_3)$$
$$\rightarrow (\neg z_1 \wedge \neg z_2 \wedge \neg z3)))))))$$
$$(e_{11} \wedge \neg e_4 \wedge \neg e_{12} \wedge \neg e_3 \wedge \neg e_2)$$
$$\rightarrow (z_1 \wedge \neg z_2 \wedge \neg z_3 \wedge X((e_4 \wedge \neg e_{11} \wedge \neg e_{12} \wedge \neg e_3 \wedge \neg e_2)$$
$$\rightarrow (z_3 \wedge \neg z_1 \wedge \neg z_2)))$$

Как видно из приведенного примера, недостатком данного подхода является сильное увеличение формулы, задающей спецификацию системы, а именно на $(|I| + |O|) \cdot |SC|$, где $|I|$ — размер множества переменных среды, $|O|$ — размер множества переменных системы, а $|SC|$ — количество элементов сценариев в заданных примерах поведения. Из-за увеличения формулы растёт количество времени, необходимого для построения автомата ко-Бюхи, также непосредственно увеличивается сам автомат ко-Бюхи, что в итоге ведёт к неэффективному разрастанию соответствующей булевой формулы и увеличению времени, затраченного на синтез целевой автоматной системы. Кроме этого в данном методе никак не учитываются особенности примеров поведения.

2.2. Граф сценариев

Другим способом представления примеров поведения системы является представление в виде дерева или графа сценариев. Оригинальный алгоритм построения дерева сценариев был описан в [5], в данной работе будет использоваться его модифицированная версия. Дерево или граф сценариев представляет из себя дерево или соответственно граф, где на каждом ребре написано состояние переменных среды и ожидаемое состояние переменных системы. При обходе дерева или графа с помощью проекции условий на ребрах можно получить все сценарии из списка сценариев, на основе которого оно или он построены. Изначально дерево содержит лишь одну вершину — корень,

далее последовательно добавляются элементы сценариев по следующему принципу: если из текущей вершины есть ребро по условию, представленному в элементе сценария, то выполняется переход по данному ребру в следующую вершину, если такого ребра нет, то добавляется новая вершина, в которую из текущей вершины проводится новое ребро с необходимым условием, после чего выполняется переход в созданную вершину.

Для примера допустим, что у нас есть следующие сценарии:
$(e_2|\,)$
$(e_2|\,) \rightarrow (e_{11}|z_1) \rightarrow (e_2|\,) \rightarrow (e_{12}|z_2) \rightarrow (e_2|\,)$
$(e_3|z_1) \rightarrow (e_2|\,) \rightarrow (e_{12}|z_2) \rightarrow (e_2|\,)$
$(e_2|\,) \rightarrow (e_{11}|z_1) \rightarrow (e_2|\,) \rightarrow (e_{12}|z_2) \rightarrow (e_3|z_1) \rightarrow (e_2|\,) \rightarrow (e_{12}|z_2) \rightarrow (e_2|\,)$
$(e_3|z_1) \rightarrow (e_2|\,) \rightarrow (e_{12}|z_2) \rightarrow (e_3|z_1) \rightarrow (e_2|\,) \rightarrow (e_{12}|z_2) \rightarrow (e_2|\,)$
$(e_4|z_3)$
$(e_2|\,) \rightarrow (e_{11}|z_1) \rightarrow (e_4|z_3)$
$(e_3|z_1) \rightarrow (e_4|z_3)$

Дерево сценариев для заданных примеров поведения, построенное по описанному выше алгоритму, представлено на рис. 4.

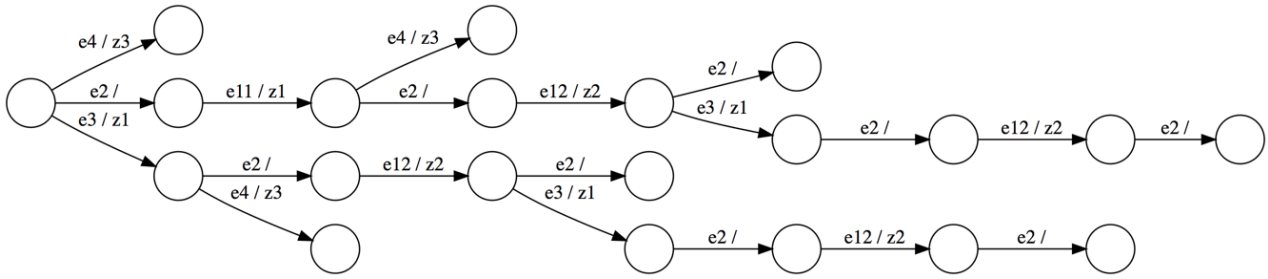

Рисунок 4 — Пример дерева сценариев

Как видно из данного примера дерево сценариев позволяет более эффективно задавать примеры поведения, сокращая количество информации, необходимой для представления сценариев.

Разовьём идею объедения примеров поведения в дерево сценариев и перейдем к графу сценариев. Граф сценариев отличается от дерева тем, что теперь кроме объединения начальных элементов сценариев, также проводится

слияние конечных элементов сценариев. Процесс построения графа сценариев начинается также, как и построение дерева, однако теперь для каждой вершины также запоминаются обратные ребра. После окончания первого этапа все вершины, из которых не исходит никаких ребер объединяются в одну конечную вершину. Далее начиная с конечной вершины происходит рекурсивный обход графа по обратным ребрам таким образом, что если из вершины исходит несколько обратных ребер с одинаковым условием, то вершины, в которые ведут эти ребра, сливаются в одну, после чего выполняется переход эти вершины и процесс повторяется заново, пока не дойдет до корня. Стоит отметить один ключевой пункт этого алгоритма, при обходе и слиянии вершин по обратным ребрам очень важно не допустить ситуации, когда мы добавляем возможность получить новые сценарии, которых не было в изначальном списке.

Для лучшего понимания приведем пример такой ситуации, допустим после первого шага мы имеем дерево, представленное на рис. 5.

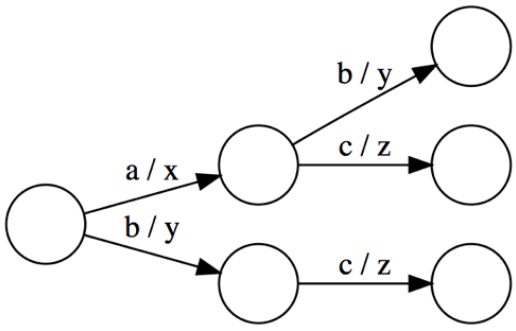

Рисунок 5 — Дерево сценариев, при слиянии вершин которого произойдет конфликт путей

Заметим, что если не учитывать случай, описанный выше, то при слиянии вершин по обратным ребрам получится следующий граф сценариев:

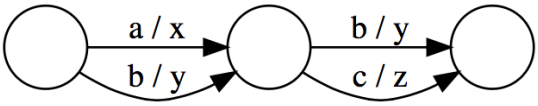

Рисунок 6 — Граф сценариев, допускающий непредусмотренные переходы

В графе, приведенном на рис. 6, выводится сценарий $(b|y) \rightarrow (b|y)$, которого не было в исходном списке сценариев. Это может привести к синтезу неверной автоматной системы и вызвать проблемы с поиском или даже отсутствие решения в задаче о выполнимости булевой формулы, если формулы линейной темпоральной логики в спецификации запрещают или противоречат таким переходам. Чтобы избежать описанных проблем можно воспользоваться подходом, применяемом в алгоритме минимизации конечных автоматов, с помощью которого вершины разбиваются на классы эквивалентности и затем по ним строится новый граф сценариев.

В качестве последнего примера на рис. 7 приведем граф сценариев для тех же примеров поведения, для которых выше было построено дерево сценариев.

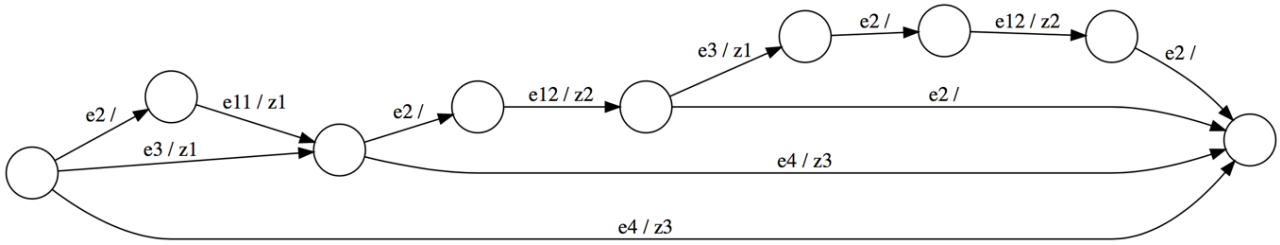

Рисунок 7 — Пример графа сценариев

Использование графа сценариев позволяет ещё больше сократить количество информации, необходимой для представления примеров поведения. Описанные далее в настоящей главе методы будут так или иначе основываться на поиске сопоставления вершин дерева или графа сценариев вершинам целевой системы переходов.

2.3. Методы учета примеров поведения при сведение к задаче SAT

Как было сказано выше, в основе предложенных методов учета примеров поведения при синтезе автоматных моделей будет лежать поиск сопоставления между вершинами графа сценариев и состояниями системы переходов. Для этого введем новый тип булевых переменных, задающих

соответствие вершин и состояний $s_{t,j}$. Таким образом $s_{t,j}$ истинна тогда и только тогда, когда состояние t системы переходов соответствует вершине j графа сценариев. Также обозначим за $out(j)$ множество исходящих ребер из вершины j, представленных в виде троек из вершины, в которую входит ребро, и условий на состояние переменных, контролируемых средой, и переменных, контролируемых системой.

Для подтверждения правильности установленного соответствия между вершинами графа сценариев и состояниями системы переходов необходимо для каждой вершины проверить соответствие между исходящими из неё ребрами и переходами из системы переходов. Другими словами, если состояние t соответствует вершине j, то для каждого исходящего из текущей вершины ребра с условиями $\boldsymbol{i}$ и $\boldsymbol{o}$ в вершину j' должен существовать переход в системе переходов из состояния t в состояние t' при состоянии переменных среды равным вектору $\boldsymbol{i}$ и переменных системы равным вектору $\boldsymbol{o}$ и при этом состояние t' должно соответствовать вершине j'. Предикат, задающий данное условие, представлен ниже:

$$\bigwedge_{t\in T}\bigwedge_{j\in ST} s_{t,j} \rightarrow \bigwedge_{(j',\boldsymbol{i},\boldsymbol{o})\in out(j)}\bigvee_{t'\in T}(\tau_{t,\boldsymbol{i},t'} \wedge \boldsymbol{o}_{t,\boldsymbol{i}} \wedge s_{t',j'})$$

где переменная $\tau_{t,\boldsymbol{i},t'}$ из оригинальной кодировки задает переход из состояния t состояние t' при переменных среды равных вектору $\boldsymbol{i}$, а вектор переменных $\boldsymbol{o}_{t,\boldsymbol{i}}$ задает состояние переменных среды при выполнении данного перехода равным вектору $\boldsymbol{o}$.

Эту формулу можно использовать не только для проверки корректности данного соответствия, но и для его поиска, передав её в программу для поиска назначения переменных, при которых формула выполнится. Единственным дополнительным условием, которое необходимо добавить в формулу является s_{t_0,j_0}, утверждающее соответствие начального состояния системы переходов и корневой вершины графа сценариев.

Размер полученной формулы составляет $\mathcal{O}(n^2 \cdot |SG|^2 \cdot |O|)$ дизъюнктов, и она содержит $\mathcal{O}(n \cdot (2^{|I|} \cdot |O| + |SG|))$ переменных, где $n = |T|$ — размер системы переходов, $|SG|$ — размер графа сценариев, $|I|$ и $|O|$ размеры множеств переменных, контролируемых средой и системой соответственно. Предполагается, что данный предикат будет дописан к оригинальной формуле, представленной в [2], что увеличит количество переменных, входящих в исходную формулу на $n \cdot |SG|$.

Однако стоит отметить, что в данный момент предложенная формула никак не использует предикат из оригинальной формулы, утверждающий, что для каждого состояния из него существует переход для всех вариантов состояний переменных среды.

$$\bigwedge_{t \in T} \bigwedge_{\boldsymbol{i} \in 2^I} \bigvee_{t' \in T} \tau_{t,\boldsymbol{i},t'}$$

Для этого модифицируем нашу формулу, поменяв условие, задающее соответствие переходов в графе сценариев и в системе переходов. Данная модификация меняет смысл исходной формулы. Теперь, если состояние t соответствует вершине j, то для каждого исходящего из заданной вершины ребра с условиями $\boldsymbol{i}$ и $\boldsymbol{o}$ в вершину j', если переход в системе переходов ведет из состояния t в состояние t' при состоянии переменных среды равным вектору $\boldsymbol{i}$, то состояние переменных системы должно быть равным вектору $\boldsymbol{o}$ и состояние t' должно соответствовать вершине j'. Модифицированная формула имеет следующий вид:

$$\bigwedge_{t \in T} \bigwedge_{j \in ST} s_{t,j} \rightarrow \bigwedge_{(j',\boldsymbol{i},\boldsymbol{o}) \in out(j)} \bigwedge_{t' \in T} (\tau_{t,\boldsymbol{i},t'} \rightarrow (\boldsymbol{o}_{t,\boldsymbol{i}} \wedge s_{t',j'}))$$

Полагается, что обновленная формула позволит более быстрее находить верное решение или факт отсутствия такового.

2.4. Кластеризованный граф сценариев

Перед переходом к методам, основанным на сведении к задаче о выполнимости булевой формулы с кванторами, стоит остановиться на

модификации графа сценариев, а именно разбиении его на кластеры. Данный подход нацелен на ещё более сильное сжатие информации о примерах поведения, что ведёт к упрощению получаемой булевой формулы. Рассмотрим два подхода к кластеризации — вершинная кластеризация и глобальная кластеризация.

2.4.1. Вершинная кластеризация

Первым рассмотрим подход вершинной кластеризации, так как он предполагает меньшее изменение графа сценариев. Он заключается в том, что условия на ребрах графа сценариев делятся на условия для переменных, контролируемых средой, и на условия, контролируемые системой. Далее ребра, исходящие из вершины, разбиваются на кластеры по условиям на переменные среды, для этого добавляется новая мнимая вершина, в которую из исходной вершины ведет ребро с условием на переменные среды и из которой исходят ребра с условиями на переменные контролируемые системой, ассоциированные с этим кластером. Для кластеров, в которых находится лишь одно ребро, новая вершина не добавляется.

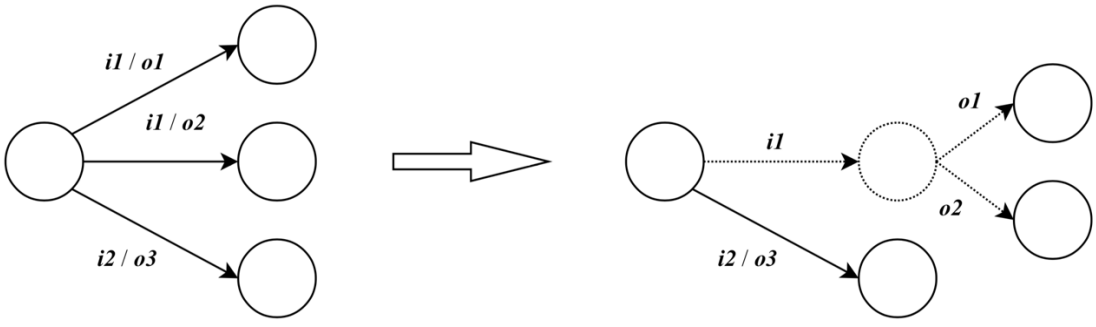

Рисунок 8 — Построение вершинных кластеров

На рис. 8 изображен пример построения вершинной кластеризации, пунктирной линией выделена вспомогательная мнимая вершина, необходимая для разбиения на кластеры.

2.4.2. Глобальная кластеризация

Теперь рассмотрим второй подход. В отличии от вершинной кластеризации при использовании метода глобальной кластеризации предполагается разбиение на кластеры сразу всех рёбер в графе. Для этого, как и в предыдущем варианте, условия на ребрах разбиваются по типу переменных. Далее, используя все ребра графа, строятся глобальные кластеры, ассоциированные с условиями на переменные, контролируемые средой. После чего ребра, находящиеся внутри таких кластеров, снова разбиваются на подкластеры на этот раз связанные по условиям на переменные, контролируемые системой.

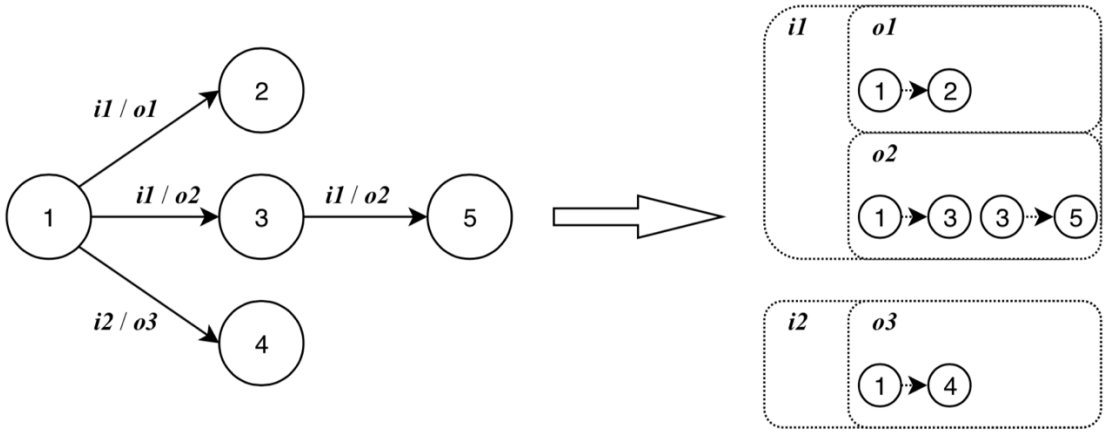

Рисунок 9 — Разбиение на глобальные кластеры

Пример разбиения графа сценариев на глобальные кластеры приведен на рис. 9. Таким образом, используя предложенные техники кластеризации можно эффективно сжать информацию, необходимую для задания примеров поведений в виде дерева или графа сценариев.

2.5. Методы учета примеров поведения при сведение к задаче QSAT

Методы учета примеров поведения при сведении к задаче о выполнимости булевой формулы имеют в основе тот же подход о поиске соответствия между графом сценариев и системой переходов, что и методы при

сведении к формуле без кванторов. В оригинальной формуле [2] основным отличием от бескванторной формулы являлся наличие квантора всеобщности по переменным среды, что позволило представить переменные $\tau_{t,t'}$ и $\boldsymbol{o}_t$ как функции от переменных $\boldsymbol{i}$. Это решение позволило сократить размер самой формулы и количество входящих в неё переменных, также благодаря этому появляется возможность более эффективного кодирования предиката, что было недоступно в SAT варианте методов, описанных ранее.

Первая версия формулы получается с помощью модификации бескванторной версии с поправкой на новые условия. Смысл нового предиката заключается в следующем: если состояние t соответствует вершине j, то для каждого исходящего из текущей вершины ребра с условиями $\boldsymbol{i}$ и $\boldsymbol{o}$ в вершину j', если в данный момент переменные среды находятся в состоянии $\boldsymbol{i}$, то должен существовать переход в системе переходов, который ведет из состояния t в состояние t', состояние переменных системы должно быть равным вектору $\boldsymbol{o}$ и состояние t' должно соответствовать вершине j'. Новая формула выглядит следующим образом:

$$\bigwedge_{t\in T}\bigwedge_{j\in SG} s_{t,j} \rightarrow \bigwedge_{(j',\boldsymbol{i},\boldsymbol{o})\in out(j)}\bigvee_{t'\in T}(\boldsymbol{i} \rightarrow (\tau_{t,t'} \wedge \boldsymbol{o}_t \wedge s_{t',j'}))$$

Сразу же можно отметить, что переменные $\boldsymbol{i}$ и $\boldsymbol{o}_t$, стоящие под оператором $\bigvee_{t'\in T}$, не зависят от t', поэтому их можно вынести из-под его действия без каких-либо потерь.

$$\bigwedge_{t\in T}\bigwedge_{j\in SG} s_{t,j} \rightarrow \bigwedge_{(j',\boldsymbol{i},\boldsymbol{o})\in out(j)} \boldsymbol{i} \rightarrow \boldsymbol{o}_t \wedge \bigvee_{t'\in T}(\tau_{t,t'} \wedge s_{t',j'})$$

Размер первой версии формулы составляет $\mathcal{O}(n \cdot |SG|^2 \cdot (|O| + |I| + n))$ дизъюнктов, а количество новых переменных всё также составляет $\mathcal{O}(n \cdot |SG|)$.

Теперь перейдем ко второму варианту предиката для задачи о выполнимости булевой формулы с кванторами. Во второй версии формулы предлагается использовать предложенный ранее принцип глобальной кластеризации. Для этого разобьем данный граф на кластеры по переменным

среды, обозначим множество векторов, характеризующих кластеры, как IC. Также теперь множество $out(j, \boldsymbol{i})$ содержит только те исходящие из вершины j ребра, которые лежат в кластере для вектора переменных среды $\boldsymbol{i}$. Множество вершин, находящихся в соответствующем кластере, обозначим как $ST(\boldsymbol{i})$. Если переменные среды находятся в состоянии $\boldsymbol{i}$ и вершине j соответствует состояние t, то для каждого ребра, исходящего из этой вершины и лежащего в кластере, ассоциированном с вектором $\boldsymbol{i}$, с условием $\boldsymbol{o}$ в вершину j', должен существовать соответствующий переход в системе переходов.

$$\bigwedge_{\boldsymbol{i}\in IC} \boldsymbol{i} \to \bigwedge_{t\in T} \bigwedge_{j\in SG(\boldsymbol{i})} s_{t,j} \to \bigwedge_{(j',\boldsymbol{o})\in out(j,\boldsymbol{i})} \boldsymbol{o}_t \wedge \bigvee_{t'\in T} (\tau_{t,t'} \wedge s_{t',j'})$$

Внешний оператор $\bigwedge_{\boldsymbol{i}\in IC}$ фиксирует состояние переменных среды, что должно облегчить проверку и поиск зависимости для переменных $t_{t,t'}$ и $\boldsymbol{o}_t$. Размер второй версии формулы равен $\mathcal{O}(|IC| \cdot (|I| + n \cdot |SG|^2 \cdot (|O| + n)))$ дизъюнктов. Данную версию формулы выгодно использовать при относительно небольших размерах $|IC|$, в противном случае она может привести к увеличению времени необходимого для поиска верного решения.

Для третьего варианта формулы мы, как и в случае с бесквантором формулой, используем предикат полноты функции переходов из оригинальной формулы.

$$\bigwedge_{t\in T} \bigvee_{t'\in T} \tau_{t,t'}$$

Также в текущей версии формулы для большей эффективности будет применен подход вершинной кластеризации, позволяющий более эффективно задавать граф сценариев, при этом не страдая от случаев с большим количеством кластеров. Обозначим множество векторов, задающих состояние переменных среды, ассоциированных с кластерами для вершины j как $IC(j)$.

$$\bigwedge_{t\in T} \bigwedge_{j\in SG} s_{t,j} \to \bigwedge_{\boldsymbol{i}\in IC(j)} \boldsymbol{i} \to \bigwedge_{(j',\boldsymbol{o})\in out(j,\boldsymbol{i})} \boldsymbol{o}_t \wedge \bigwedge_{t'\in T} (\tau_{t,t'} \to s_{t',j'})$$

Асимптотика новой формулы аналогична первой версии и составляет $\mathcal{O}(n \cdot |SG|^2 \cdot (|O| + |I| + n))$, однако несмотря на это ожидается, что эта

формула будет более эффективна, благодаря более эффективной работе с ограничениями на переменные $\tau_{t,t'}$, задающими функцию переходов целевой системы переходов, и использовании вершинной кластеризации.

Выводы по главе 2

Во второй главе были введены новые методы учета примеров поведения при синтезе автоматных систем по спецификации, заданной в виде темпоральных формул. Также были описаны алгоритмы построения графов сценариев и способы разбиение графов на кластеры, призванные повысить эффективность представления примеров поведения, используемого при кодировании примеров поведения в виде булевых формул.

ГЛАВА 3. АНАЛИЗ РЕЗУЛЬТАТОВ

В настоящей главе будет произведено сравнение предложенных методов учета примеров поведения при синтезе автоматных моделей по темпоральным формулам, а также будет приведен анализ результатов. Тестирование выполнялось на компьютере с операционной системой Ubuntu 16.04, процессором Intel Core i5 5257U и 4 гигабайтами оперативной памяти. Преобразование линейных темпоральных свойств в автомат ко-Бюхи производилось с помощью утилиты spot, в качестве программного средства для поиска решения задачи SAT использовался cryptominisat, в качестве средства для решения задачи QSAT — rareqs. Для обработки входных данных использовалась модифицированная версия программы BoSy [11], представленная авторами [2], написанная на языке программирования Swift, в которую были встроены предложенные в работе методы. Также использовались скрипты на языке Python для генерации примеров поведения и представления сценариев в виде темпоральных формул.

3.1. Тестирование методов учета примеров поведения

В качестве тестов использовалась часть тестового набора сущностей, представленных на соревновании по синтезу реактивных систем SYNTCOMP [12]. Для проверки эффективности методов учета примеров поведения с помощью программы, написанной на языке Python, для каждой сущности были сгенерированы случайные примеры поведения. Генерация проводилась следующим образом: на вход программе подавалась автоматная система, полученная по спецификации конкретной сущности, для которой было необходимо построить примеры поведения, представленной в виде ориентированного графа с условиями на переменные среды и переменные системы, указанных на переходах, и число сценариев, которые необходимо сгенерировать. Далее для каждого сценария случайно определялась его длина n в диапазоне от 2 до $|\text{размер автоматной системы}| \cdot 5$. После этого, стартуя в начальном состоянии каждый раз случайным образом выбирается исходящий

переход в следующее состояние и к сценарию дописывается условие на переменные данного перехода. Если существует несколько назначений переменных среды, удовлетворяющих условие перехода, то случайно выбирается одно из них. После того как была сгенерирована новая спецификация в виде линейных темпоральных формул и примеров поведения, она передавалась на вход другой утилите, которая преобразовывала примеры поведения в формулы линейной темпоральной логики для сравнения с базовой реализацией. Полученные в итоге данные подавались на вход тестируемой программе.

При тестировании замерялось время работы каждого отдельного шага алгоритма bounded synthesis, а также суммарное время работы всех шагов. Для каждых входных данных и каждого метода производилось порядка двадцати запусков, среди которых выбиралось среднее время на каждом этапе, также перед этим предварительно выполнялось по пять запусков без замеров времени.

3.2. Сравнение результатов

В данный момент стоит вспомнить порядок работы подхода bounded synthesis, в частности его первый шаг, который заключается в преобразовании формул линейной темпоральной логики в автомат ко-Бюхи. При выражении примеров поведения в виде темпоральных формул они наряду с формулами спецификации будут переведены в автомат ко-Бюхи, также стоит отметить, что данные формулы обычно довольно громоздкие. Из-за данных факторов ожидается, что первый шаг алгоритма будет занимать гораздо больше времени, так как необходимо разрешить большее количество сложных формул. Кроме того, полученный автомат ко-Бюхи должен получаться большим и неоптимальным из-за чего также будет расти количество переменных, задающих функцию аннотации. В результате приведенных выше замечаний размер итоговой сгенерированной формулы заметно увеличится, и поиск решений займет большее количество времени.

На соревновании по синтезу автоматных моделей SYNTCOMP [12] представлено множество различных сущностей, однако все их можно разбить на группы по классам решаемых задач. В итоговый набор сущностей, используемый для тестирования, вошло по нескольку представителей каждого из этих классов. Примеры поведения для сущностей из настоящего набора были сгенерированы по алгоритму, описанному в предыдущем пункте.

Результаты проведенных экспериментов для подходов, основанных на сведении к задаче SAT, и подходов, основанных на сведении к QSAT, представлены на графиках, изображенных на рис. 10 и рис. 11 соответственно. Вертикальная шкала в графиках, представляет время в секундах, а горизонтальная количество сущностей, для которых было найдено решение. Время работы для каждой сущности определялось, как суммарное время работы всех этапов алгоритма bounded synthesis.

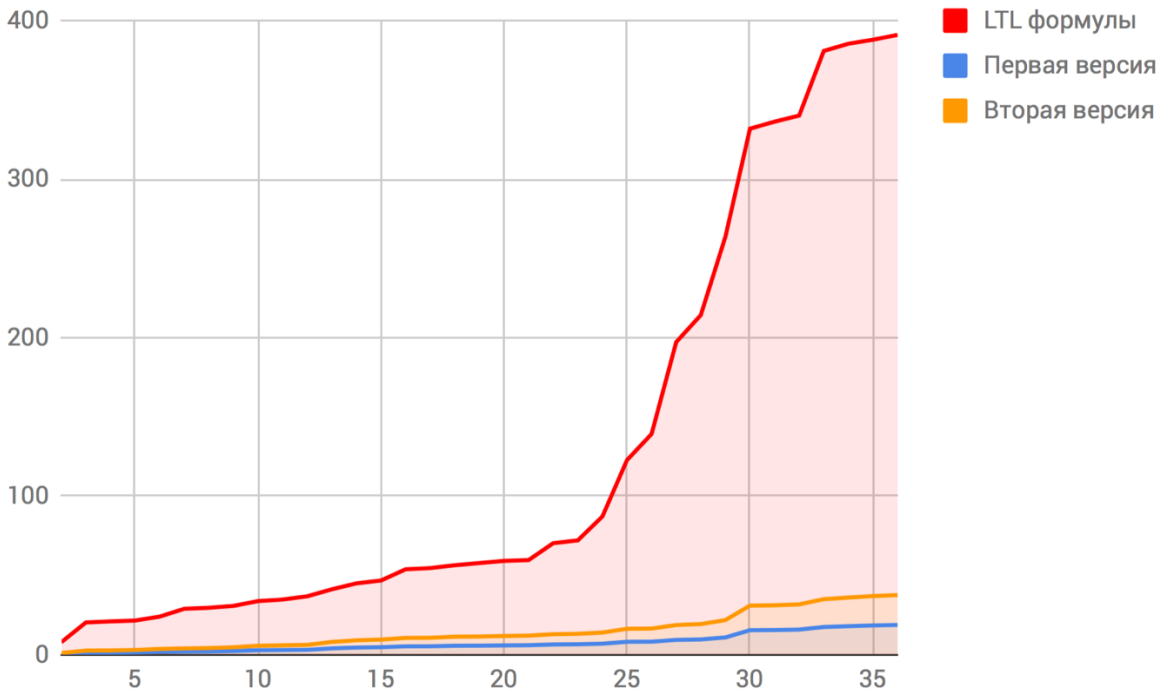

Рисунок 10 — Сравнение скорости работы различных методов при сведении к задаче SAT

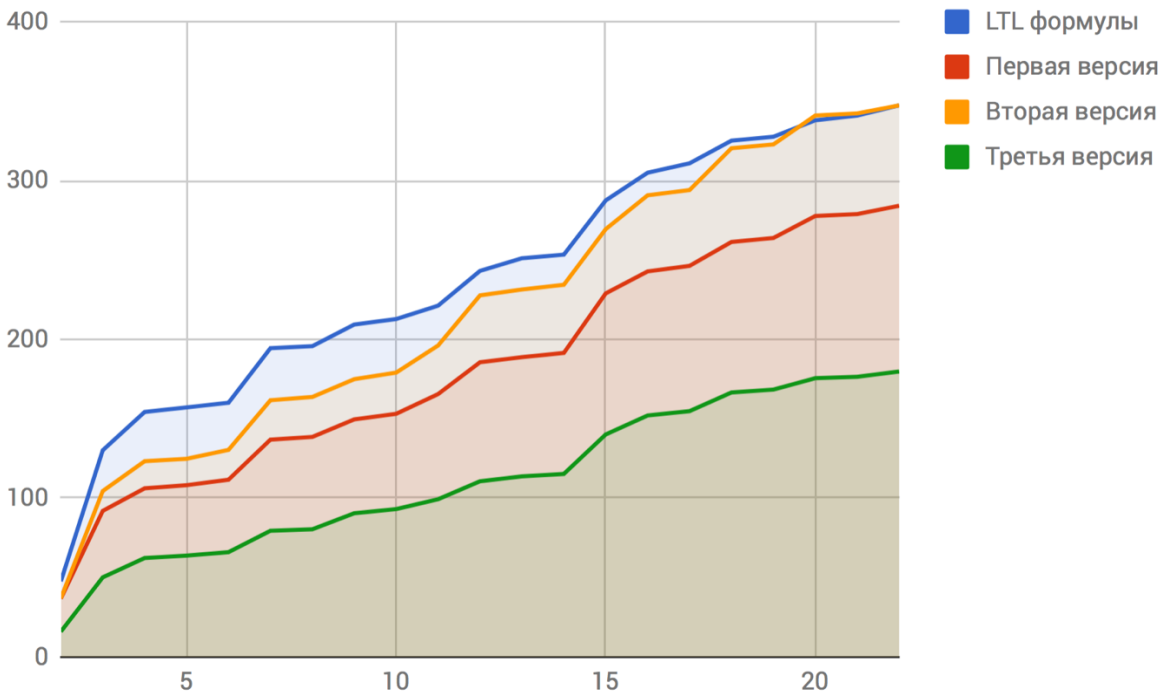

Рисунок 11 — Сравнение скорости работы различных методов при сведении к задаче QSAT

Теперь рассмотрим подробнее результаты нескольких тестов, разбитых по шагам работы подхода bounded synthesis.

В таблице 1 представлены результаты экспериментов для методов, основанных на сведении к задаче SAT. Данные результаты показывают среднее время работы каждого этапа алгоритма bounded synthesis при использовании советующего метода представления примеров поведения на одной и той же спецификации.

Таблица 1 — Сравнение результатов работы SAT кодировок

Использованный метод	Первая версия	Вторая версия	LTL формулы
Время построения автомата ко-Бюхи, с	0.1	0.1	5.82
Время генерации формулы, с	0.9	0.86	2
Время решения формулы, с	2	2.6	11.7
Общее время, с	3	3.56	19.52

Как видно из таблицы 1 и графика, изображенного на рис. 10, для варианта при сведении задачи к задаче о выполнимости булевой формулы лучшим образом себя показала первая версия формулы, хотя стоит отметить, что вторая версия не сильно от неё отстала. Такой большой прирост оправдан тем, что в кодировке для SAT версии bounded synthesis используется перебор по $2^{|I|}$ переменных и с увеличением автомата ко-Бюхи формула очень сильно разрастается. В предложенных же методах нет зависимости от $2^{|I|}$, поэтому формула не так сильно увеличиваются при росте графа сценариев.

Аналогично, в таблице 2 и в таблице 3 приведены результаты экспериментов для различных методов учета примеров поведения при сведении к задаче QSAT. Так же, как и в таблице 1, в настоящих таблицах приведены средние показатели на каждом этапе работы алгоритма при использовании советующего подхода на одной и той же спецификации для каждой таблицы.

Таблица 2 — Сравнение результатов работы QSAT кодировок

Использованный метод	Первая версия	Вторая версия	Третья версия	LTL формулы
Время построения автомата ко-Бюхи, с	0.014	0.018	0.015	3.24
Время генерации формулы, с	0.1	0.11	0.1	0.17
Время решения формулы, с	4.8	6.4	3.4	5
Общее время, с	4.914	6.528	3.515	8.41

По приведенным в таблицах 2 и 3, а также на графике, изображенном на рис. 11, результатам видно, для QSAT кодировок, лидером среди которых является третья версия кодировки, использующая предикат полноты функции переходов и вершинную кластеризацию, прирост производительности также является существенным, хотя и не таким большим как для SAT версий.

Таблица 3 — Сравнение результатов работы QSAT кодировок

Использованный метод	Первая версия	Вторая версия	Третья версия	LTL формулы
Время построения автомата ко-Бюхи, с	0.17	0.16	0.14	5.43
Время генерации формулы, с	0.35	0.37	0.37	0.55
Время решения формулы, с	74.9	89.2	45.3	78.8
Общее время, с	74.42	89.73	45.81	84.78

Также после проведения экспериментов подтвердился тот факт, что при представлении примеров поведения в виде формул линейной темпоральной логике из-за усложнения структуры LTL спецификации существенно увеличивается время, необходимое для построения автомата ко-Бюхи, что хорошо видно в показателях, приведенных в таблице 1, таблице 2 и таблице 3.

Выводы по главе 3

В третьей главе было проведено экспериментальное исследование разработанных в предыдущей главе методов учета примеров поведения и базовых подходов к представлению примеров поведения. Результаты экспериментов показали превосходство новых разработанных методов относительно представления сценариев в виде темпоральных формул, как в случае со сведением к задаче SAT, так при сведении к задаче QSAT. В случае с кодировкой для задачи SAT новые методы демонстрируют многократный прирост производительности.

ЗАКЛЮЧЕНИЕ

В данной работе были разработаны новые методы учета примеров поведения при синтезе автоматных моделей по темпоральным формулам на основе подхода синтеза автоматных моделей с ограничением на размер целевой системы. Также были предложены новые варианты представления примеров поведения в виде кластеризованных графов сценариев. Кроме того, было проведено экспериментальное исследование, показавшее высокую эффективность новых методов и кратном превосходстве над базовыми методами.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Ulyantsev, V., Buzhinsky, I. & Shalyto, A. Exact finite-state machine identification from scenarios and temporal properties // International Journal on Software Tools for Technology Transfer. — 2018. — Т. 20, № 1. — С. 35-55.
2. Peter Faymonville, Bernd Finkbeiner, Markus N. Rabe, Leander Tentrup. Encodings of Bounded Synthesis // Tools and Algorithms for the Construction and Analysis of Systems. — 2017. — С. 354-370.
3. Ulyantsev V., Zakirzyanov I., Shalyto A. Symmetry Breaking Predicates for SAT-based DFA Identification. — 2016.
4. The 4th Reactive Synthesis Competition (SYNTCOMP 2017): Benchmarks, Participants & Results / Swen Jacobs [и др.] // Sixth Workshop on Synthesis (SYNT 2017). Electronic Proceedings in Theoretical Computer Science. — 2017. — С. 116-143.
5. Ulyantsev, V., Tsarev, F. Extended finite-state machine induction using SAT-solver // IFAC Proceedings Volumes. — 2012. — Т. 45, № 6. — С. 236–241.
6. Finkbeiner B., Schewe S. SMT-based synthesis of distributed systems // Second Workshop on Automated Formal Methods. — 2007.
7. Heule M.J., Verwer S. Software model synthesis using satisfiability solvers // Empirical Software Engineering. — 2013. — Т. 18, № 4. — С. 825–856.
8. Finkbeiner B., Schewe S. Bounded synthesis // International Journal on Software Tools for Technology Transfer. — 2013. — Т. 15, № 5-6. — С. 519–539.
9. Clarke E.M., Grumberg O., Peled D. Model checking. — США MIT press. — 1999.
10. Eén N., Sörensson N. Temporal induction by incremental SAT solving // Electronic Notes in Theoretical Computer Science. — 2003. — Т. 89, № 4. — С. 543–560.

11. BoSy. Reactive synthesis tool based on constraint solving. — 2016. — URL: https://github.com/reactive-systems/bosy (дата обращения: 01.03.2018)
12. EDACC Web Frontend. Syntcomp 2017. Instances. — 2017. — URL: http://syntcomp.cs.uni-saarland.de/syntcomp2017/experiment/12/instances (дата обращения: 07.05.2018)