На правах рукописи

Ушакова Мария Сергеевна

МЕТОДЫ И ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА ФОРМАЛЬНОЙ ВЕРИФИКАЦИИ ФУНКЦИОНАЛЬНО-ПОТОКОВЫХ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ

Специальность 2.3.5 — Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

АВТОРЕФЕРАТ
диссертации на соискание учёной степени
кандидата технических наук

Красноярск 2022

Работа выполнена в Федеральном государственном автономном образовательном учреждении высшего образования «Сибирский федеральный университет».

Научный руководитель: доктор технических наук, профессор, **Легалов Александр Иванович**;

Официальные оппоненты: **Водяхо Александр Иванович**, доктор технических наук, профессор, Федеральное государственное автономное образовательное учреждение высшего образования Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина), кафедра вычислительной техники, профессор;

Феоктистов Александр Геннадьевич, кандидат технических наук, доцент, Федеральное государственное бюджетное учреждение науки Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук, Лаборатория параллельных и распределённых вычислительных систем, заведующий лабораторией.

Ведущая организация: Федеральное государственное бюджетное учреждение науки Институт систем информатики им. А.П. Ершова Сибирского отделения Российской академии наук.

Защита состоится «12» мая 2022 года в 16 часов на заседании диссертационного совета 24.2.404.05, созданного на базе Сибирского федерального университета, по адресу: 660074, г. Красноярск, ул. Академика Киренского, 26, ауд. УЛК 112.

С диссертацией можно ознакомиться в научной библиотеке и на сайте «Сибирского федерального университета» www.sfu-kras.ru.

Автореферат разослан «___» _____ 2022 г.

Учёный секретарь диссертационного совета  Кайзер Юрий Филиппович

Общая характеристика работы

Актуальность темы и степень её разработанности. Постоянный рост объёма разрабатываемых программных систем и их ориентация на параллельную обработку данных ведут к увеличению сложности программного обеспечения, что, в свою очередь, повышает вероятность возникновения различных ошибок. Традиционный способ обеспечения надёжности программ путём тестирования не может полностью удовлетворить возрастающие требования практического использования. Альтернативой тестированию являются методы формальной верификации, которые обеспечивают анализ всех возможных вариантов поведения системы. Формальная верификация — это формальное доказательство того, что программа соответствует своей спецификации. Методы формальной верификации позволяют доказать отсутствие ошибок в программе, в то время как тестирование лишь выявляет ошибки, но не даёт гарантии их отсутствия. Существуют разнообразные подходы к формальной верификации программ. Основными являются методы проверки моделей, дедуктивный анализ, различные варианты уточнения программ. Наибольший вклад в развитие формальной верификации внесли труды таких ученых, как Р. Флойд, Т. Хоар, Э. Дейкстра, Б. Мейер, К. Морган, Э. Денни, Дж. Гизль, В.А Непомнящий, О.М. Рякин, М. Кларк, Ю.Г. Карпов, И.С. Ануреев, И.В. Марьясов, В.И. Шелехов.

В настоящее время достигнуты определённые успехи в практическом применении формальной верификации к последовательным программам. Разработан ряд систем для поддержки этого процесса. В качестве примера можно привести верификаторы программ на языке Си: Boogie и система СПЕКТР; системы для верификации объектно-ориентированных программ на Java: LOOP и KeY; системы для верификации функциональных программ: DSolve, HSolve.

Параллельное программирование позволяет существенно увеличить производительность на современных вычислительных системах. Однако параллелизм приводит к значительному усложнению разработки, а особенно отладки и верификации программ. Непосредственная ориентация современного параллельного программирования на существующие архитектуры затрудняет формальную верификацию, так как наряду с логической корректностью программ приходится исследовать и влияние ресурсных ограничений, порождающих дополнительные ошибки. Ситуация может быть улучшена, если при разработке параллельных программ на первоначальных этапах можно было бы избавиться от ресурсных конфликтов. Одним из таких подходом является архитектурно-независимое параллельное программирование. Представитель данного направления — функционально-потоковая модель параллельных вычислений и созданный на её основе язык программирования Пифагор. Основными идеями подхода являются исключение ресурсных ограничений и неявное управление вычисле-

ниями. В функционально-потоковой параллельной (ФПП) программе описываются только информационные связи между выполняемыми функциями. Взаимодействие между функциями при этом осуществляется по готовности данных, что позволяет создавать программы с неограниченным параллелизмом, «сжатие» которого к конкретным вычислительным ресурсам и условиям эксплуатации будет происходить после верификации и отладки написанных программ.

Перспективным видится то, что в отсутствии ресурсных ограничений основной упор в разработке параллельных программ и их анализе можно сделать именно на формальную верификацию при отсутствии ресурсных конфликтов. Вместе с тем следует отметить, что вопросы, связанные с формальной верификацией, в рамках данного направления, практически не проработаны. Имеются работы, связанные с организацией отладки ФПП программ и использованием методов верификации для анализа корректности данных, не связанных с формальным доказательством логической корректности кода.

Поэтому актуальной является разработка формальных методов и средств верификации для языка функционально-потокового параллельного программирования, обеспечивающих проверку логики функционирования программ.

Цель и задачи исследования. Целью работы является повышение надёжности и корректности функционально-потоковых параллельных программ посредством разработки методов формальной верификации.

Для достижения указанной цели в работе решаются следующие задачи:

— проанализировать существующие подходы к формальной верификации и возможности их использования при функционально-потоковом параллельном программировании;

— формально описать семантику языка ФПП программирования Пифагор для формирования основы методов анализа ФПП программ;

— разработать методы, обеспечивающие формальную верификацию ФПП программ;

— разработать инструментальное средство, обеспечивающее поддержку методов формальной верификации ФПП программ.

Объект исследования. Функционально-потоковая модель параллельных вычислений и язык функционально-потокового параллельного программирования.

Предмет исследования. Методы и средства формальной верификации функционально-потоковых параллельных программ.

Научная новизна исследования.

1. Для доказательства корректности функционально-потоковых параллельных программ, написанных на языке Пифагор, впервые разработан метод верификации на базе исчисления Хоара, эквивалентный по сложности методам доказательства корректности для последовательных программ, но позволяю-

щий доказывать корректность программы без ограничения параллелизма, что обеспечивается концепцией неограниченных вычислительных ресурсов.

2. Для доказательства завершения функционально-потоковых параллельных программ впервые предложен метод, использующий ограничивающую функцию, который допускает изменение спецификации программы таким образом, чтобы доказательство частичной корректности одновременно характеризовало и завершение программы. Это позволяет строить инструментальные средства поддержки доказательства на одной общей основе.

3. Для функционально-потоковых параллельных программ впервые предложен метод удаления взаимной рекурсии нескольких функций, позволяющий преобразовывать произвольную функцию в функцию с прямой рекурсией, это повышает эффективность верификации за счёт преобразования программы только к одной функции.

4. Впервые разработана архитектура и реализован прототип инструментального средства, обеспечивающего поддержку верификации функционально-потоковых параллельных программ с помощью предложенных методов. Это позволяет упростить процесс формальной верификации за счёт наглядной визуализации дерева доказательства и автоматизации преобразований графа анализируемой функции.

Положения, выносимые на защиту.

1. Разработанный метод верификации на базе исчисления Хоара для ФПП программ на языке Пифагор позволяет доказывать частичную корректность ФПП программ.

2. Разработанный метод доказательства завершения ФПП программ совместно с методом верификации позволяет доказать тотальную корректность программы.

3. Предлагаемый метод удаления взаимной рекурсии нескольких функций позволяет преобразовать любую рекурсивную функцию ФПП программы в прямую рекурсию, чтобы применить к ней метод верификации.

4. Разработанный прототип инструментального средства для поддержки верификации ФПП программ позволяет визуализировать процесс доказательства и автоматизирует построение дерева доказательства для упрощения процесса формальной верификации.

Теоретическая и практическая значимость работы. Результаты работы могут быть использованы для повышения эффективности разработки параллельных программ. На основе предложенных методов формальной верификации разработано инструментальное средство, обеспечивающее поддержку верификации функционально-потоковых параллельных программ.

Исследование выполнено при финансовой поддержке РФФИ в рамках научных проектов № 13-01-00360, № 17-07-00288 и ФЦП «Научные и научно-

педагогические кадры инновационной России» № 14.A18.21.0396.

Полученные научные и практические результаты использованы в учебном процессе на кафедре вычислительной техники и кафедре высокопроизводительных вычислений Института космических и информационных технологий СФУ при изучении дисциплин «Технологии разработки программного обеспечения», «Параллельное программирование», «Высокопроизводительные вычисления на графических процессах» и при выполнении выпускных квалификационных работ.

Результаты исследования использованы при разработке функционально-потоковых параллельных программ для процесса высокоуровневого проектирования цифровых схем на базе функционально-потоковой парадигмы. Для разработки ФПП программ использовалась формализованная семантика языка Пифагор, проводилась формальная верификация ФПП программ перед их преобразованием в программы для сверхбольших интегральных схем.

Результаты практического применения диссертационного исследования подтверждены актами о внедрении.

Методы исследования. В работе использовались элементы теории множеств, методы математической логики (различные логики и аксиоматические теории, λ-исчисление, метод доказательства теорем на основе исчисления Хоара, методы доказательства завершения программ на базе трансфинитной индукции), методы теории рекурсивных функций (построение универсальной рекурсивной функции для удаления взаимной рекурсии нескольких функций), элементы теории графов, методы и понятия теории алгоритмов, теории языков программирования, теории языков и формальных грамматик, теория разработки трансляторов. Для описания результатов использовались UML-диаграммы, диаграммы Вирта и формы Бэкуса-Наура.

При создании программных инструментов использовались структурное и объектно-ориентированное программирование, кросс-платформенная библиотека Qt.

Апробация результатов работы. Основные положения диссертации докладывались и обсуждались на 15 конференциях и семинарах, основные из которых: Международная научная конференция «Параллельные вычислительные технологии» (Челябинск, 2013; Ростов-на-Дону, 2018), Parallel Computing Technologies International Conference (Санкт-Петербург, 2013), Международная конференция «IX Сибирский конгресс женщин-математиков» (Красноярск, 2016), Всероссийская научная конференция памяти А.Л. Фуксмана «Языки программирования и компиляторы» (Ростов-на-Дону, 2017), Международная конференция «Суперкомпьютерные дни в России» (Москва, 2018).

Получено два свидетельства о регистрации программного обеспечения (2013, 2021).

Публикации. По теме диссертации опубликовано двадцать три научные работы; из которых шесть статей в изданиях, рекомендуемых ВАК; три статьи входят в список Web of Science Core Collection; пять статей индексируется в Scopus.

Личный вклад автора. Основные результаты являются новыми и получены лично автором. Автором проведён значительный объём научных изысканий, подготовлены публикации; предложен метод верификации на базе исчисления Хоара и доказательства завершения ФПП программ на языке Пифагор; предложен метод удаления взаимных рекурсий нескольких функций в программах на языке Пифагор; разработано инструментальное средство поддержки формальной верификации ФПП программ. Научные работы, опубликованные в соавторстве с научным руководителем, заключаются в разработке метода формальной верификации ФПП программ на основе дедуктивного анализа; разработке архитектуры инструментального средства поддержки формальной верификации ФПП программ. Совместно с научным руководителем автор осуществлял постановку целей и задач и анализ полученных результатов. В совместных публикациях автора с Удаловой Ю.В. включен разработанный автором метод доказательства завершения программ на языке Пифагор. В совместных публикациях автора с Непомнящим О.В., Матковским И.В., Васильевым В.М. и Романовой Д.С. научные результаты автора являются дополнением к концепции разрабатываемого ими транслятора функционально-потоковых параллельных программ.

Структура работы. Диссертационная работа состоит из введения, 4 глав, заключения, списка литературы, включающего 188 наименований, списка сокращений и 14 приложений. Работа изложена на 157 листах машинописного текста, содержит 41 рисунок, 6 таблиц.

Содержание диссертации

Во **введении** обоснована актуальность работы, определены цель и задачи исследования, указаны применяемые в работе методы, представлены основные результаты.

В **первой главе** проводится анализ существующих методов и средств формальной верификации и их применимости для ФПП программ. Рассматриваются ошибки, характерные для ФПП программ. Определяются подходы, направленные на обнаружение этих ошибок.

Ошибки в ФПП программах определяются спецификой модели вычислений, главной особенностью которой является отсутствие ошибок, обусловленных взаимодействием ограниченных ресурсов. Поэтому, как и в последовательных программах, ошибки ФПП программ можно разделить на две группы: ошибки в семантике программы и невозможность завершения программы. Од-

нако ошибки второй группы могут быть вызваны только зацикливанием программы в результате бесконечной рекурсии, так как в языке нет частично определённых функций, и все функции возвращают результат для любого аргумента. В результате анализ корректности ФПП программ сводится к анализу ошибок, аналогичных ошибкам, возникающим в последовательных программах.

Среди множества существующих методов формальной верификации программ наиболее подходящим для верификации ФПП программ является метод дедуктивного анализа, основанный на исчислении Хоара, в котором основными объектами являются тройки Хоара. При этом считается, что программа уже написана, а спецификация корректно описывает требования к программе. Основная причина выбора исчисления Хоара состоит в том, что для верификации ФПП программ не подходят методы пошагового преобразования спецификации в программу, так как нет простого способа представить параллельность выполнения операций в описываемом алгоритме на языке спецификации. Другая причина — универсальность метода, его применимость к большому классу задач и возможность использования произвольного языка спецификации достаточной выразительности, который позволяет сформулировать условия корректности всего алгоритма, а не отдельных утверждений о программе. К недостаткам выбранного метода относится невозможность полной автоматизации процесса доказательства из-за отсутствия разрешимости достаточно выразительного языка спецификации. Другой недостаток — невозможность выразить свойства завершения программы, поэтому данное свойство требуется доказывать отдельно.

Ввиду того, что все методы доказательства завершения программ имеют одну основу, базирующуюся на трансфинитной индукции, то для доказательства завершения программы выбран метод, позволяющий расширить исчисление Хоара так, чтобы доказательство корректности тройки Хоара являлось доказательством тотальной корректности программы, то есть частичной корректности и завершения. В дальнейшем это позволит строить инструментальные средства на общей базе.

Анализ инструментальных средств поддержки формальной верификации программ на основе дедуктивного анализа показал, что все системы имеют схожую архитектуру. На вход поступает код программы со спецификацией. Система состоит из модулей. Главным в системе является модуль генерации условий корректности. Этот модуль уникальный у каждой системы и определяется используемым исчислением. Он зависит от таких ключевых характеристик, как язык программирования доказываемой программы, язык спецификации, алгоритм доказательства и правила преобразования. Данный модуль у разных систем может работать как автоматически, так и интерактивно. Получаемые от модуля условия корректности передаются модулю доказательства. Практически всегда это сторонний модуль, не связанный с языком программирования, и

разработанный для доказательства утверждений на определённом языке логики. Порядок логики определяет, автоматически или интерактивно проводятся доказательства формул. В результате работы система поддержки верификации программ выдаёт сообщение о корректности программы. В некоторых системах также предполагается наличие обратной связи, позволяющей анализировать причины возникших ошибок.

Данная архитектура может быть использована в инструментальном средстве поддержки верификации ФПП программ. Основной задачей данной работы является разработка модуля генерации условий корректности, а для доказательства теорем предполагается использование сторонних средств. Это позволяет провести декомпозицию задачи и использовать имеющиеся достижения в области автоматического и автоматизированного доказательства теорем. На данном этапе работы сгенерированные условия корректности передаются пользователю, который может использовать для их доказательства одну из существующих систем.

Разработка языка спецификации ФПП программы опирается на логику системы интерактивного поиска доказательства HOL, что в дальнейшем упростит применение систем автоматизированного доказательства теорем для доказательства условий корректности.

В результате проведённого анализа для верификации ФПП программ выбран метод, основанный на исчисления Хоара, который также позволяет доказывать завершение программы при определении ограничивающей функции. Определена общая схема архитектуры инструментального средства поддержки верификации ФПП программ, данное средство позволит упростить трудоёмкий процесс доказательства корректности программ.

Во **второй главе** строится аксиоматическая теория для верификации программ на языке Пифагор на базе исчисления Хоара. Для этого вначале формализуется семантика языка Пифагор и задаётся язык спецификации свойств ФПП программ. Предлагается наглядный способ проведения доказательства на информационном графе программы.

Семантика языка Пифагор формализуется посредством описания семантики простых и составных типов, а также семантики всех встроенных функций. Семантические правила для встроенных функций представляются в виде ориентированных деревьев. В каждой вершине дерева содержится определённое логическое выражение такое, что каждый путь от корня к листу дерева описывает один из вариантов выполнения функции. Исключением являются листья дерева, в которых содержится результат выполнения функции.

В качестве языка спецификации используется исчисление конструкций. Выбор типизированной логики высшего порядка обусловлен несколькими причинами. Логика типизированная, потому что, во-первых, функции языка Пи-

фагор являются полиморфными (имеется в виду ad-hoc-полиморфизм, когда функция применима к аргументам разного типа, который определяет тип возвращаемого результата), и в спецификации требуется указывать, какой тип аргумента приводит к какому результату; во-вторых, списки могут содержать элементы разного типа, что также требуется описывать с помощью языка спецификации. Порядок логики обуславливают списки языка Пифагор. В самом общем случае список может иметь неизвестную длину и содержать элементы неизвестного типа. Для того, чтобы описать это в спецификации, требуется использовать кванторы не только для переменных и функций, но и типов. Язык спецификации вводится как аксиоматическая теория: задаётся язык, аксиомы и правила вывода выражений. Вначале вводится система типов, необходимая для описания типов объектов языка. С помощью правил конструирования описывается синтаксис термов. Далее описывается семантика, правила вывода выражений и теории для разных типов языка. Каждая теория задаётся с помощью сигнатуры и аксиом. Система является надёжной для теоретико-множественной семантики; непротиворечивой, так как имеет стандартную модель; однако не является полной.

Для доказательства корректности ФПП программ построена аксиоматическая теория языка Пифагор. Основными объектами аксиоматической теории являются тройки Хоара, для которых вводится следующее обозначение:

$$\boxed{\varphi(x)}\ \texttt{Prog(x)} \to r\ \boxed{\psi(r)}\,,$$

где `Prog(x)` — код функции с входным аргументом x, r — результат вычисления функции, $\varphi(x)$ — предусловие для `Prog`, зависящее от аргумента x, $\psi(r)$ — постусловие для `Prog`, зависящее от результата выполнения функции (и от входного аргумента, так как от него зависит результат). В качестве аксиом используются тройки Хоара для встроенных функций языка. Встроенные функции определены для всех типов аргументов, поэтому каждой функции соответствует несколько аксиом, отличающихся типами аргументов и результатами вычислений. Введено два правила вывода. Первое — правило прямого прослеживания, для некоторой функции с кодом «$\texttt{x:F}_1\texttt{:F}$» оно имеет следующий вид:

$$\left\{\boxed{P_1(x_1)}\ \texttt{x}_1\texttt{:F} \to r\ \boxed{Q(r)}\,,\ A_1,\ A_2\right\} \quad \vdash \quad \boxed{P(x)}\ \texttt{x:F}_1\texttt{:F} \to r\ \boxed{Q(r)}\,, \qquad (1)$$

где:

$$A_1 \equiv \boxed{\varphi(x)}\ \texttt{x:F}_1 \to x_1\ \boxed{\psi(x_1)}\,, \quad A_2 \equiv (P(x) \Rightarrow \varphi(x)),$$
$$P_1(x_1) \equiv ((P(x) \Rightarrow \varphi(x)) \Rightarrow \psi(x_1)),$$

`x` — входной аргумент, $\texttt{F}_1$ — функция, применяемая непосредственно к входному аргументу, `F` — остальная часть программы, которая может содержать другие функции, применяемые непосредственно к аргументу `x`, A_1 — одна из аксиом

для функции $\mathtt{F}_1$, символ выводимости $\vdash$ обозначает, что тройка Хоара справа истинна, если истинны формулы слева. Согласно этому правилу, при применении аксиомы A_1 к функции $\mathtt{F}_1$, тройка Хоара (справа) преобразуется в новую тройку (слева) с более «короткой» программой, при этом предусловие $P(x)$ изменяется на $P_1(x_1)$, а исходный аргумент x заменяется на x_1 — результат применения функции $\mathtt{F}_1$ к x. Истинность формулы A_2 — необходимое условие применения аксиомы A_1. Доказывается, что в выражении $P_1(x_1)$ импликация может быть заменена на конъюнкцию. При последовательном применении правила прямого прослеживания происходит «сокращение» или «свёртка» программы, и после анализа всех операторов можно получить тройку Хоара с «пустой» программой: $\boxed{\mathrm{P}}\ \boxed{\mathrm{Q}}$.

Второе правило позволяет преобразовать тройку с пустой программой в формулу на языке спецификации:

$$(P \Rightarrow Q) \vdash \boxed{\mathrm{P}}\ \boxed{\mathrm{Q}}\,. \tag{2}$$

С помощью преобразований произвольных троек Хоара на основе правил вывода можно получить формулу на языке спецификации, истинность которой доказывается в рамках исчисления конструкций. Если эта формула истинна, то истинна и исходная тройка Хоара, откуда следует корректность программы. Если на некотором этапе преобразования применимы несколько аксиом, то преобразования проводятся независимо для каждой аксиомы и количество новых троек будет равно количеству применимых аксиом. В этом случае доказательство корректности программы сводится к доказательству истинности конечного числа формул. Заданная система Хоара надёжна, то есть из истинных троек можно получить только истинные тройки. Также её можно считать полной в том смысле, что любую тройку всегда можно свести к формуле на языке логики конечным числом преобразований.

В связи с тем, что программу на языке Пифагор удобно отображать в виде информационного графа, то и доказательство корректности программы также можно отобразить на графе. Процесс доказательства в этом случае заключается в разметке дуг графа формулами и преобразовании графа. Граф со всеми размеченными дугами может быть преобразован в формулу на языке спецификации. Информационный граф программы, дуги которого помечены формулами на языке спецификации называется *информационным графом с разметкой* (ИГР). Если в информационном графе размечены только входная и выходная дуги, то граф соответствует тройке Хоара, в которой программа представляется графом, разметка входной дуги есть предусловие, выходной — постусловие. На рисунке 1 приведён пример ИГР.

Над ИГР заданы следующие преобразования:

1. разметка дуги — приписывание формулы к дуге,

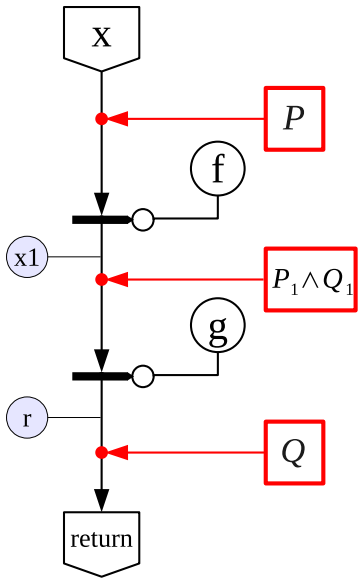

Рисунок 1 — Информационный граф с разметкой для функции с кодом «`x:f:g`», `x1` и `r` — идентификаторы дуг, P — предусловие, Q — постусловие, $P_1 \wedge Q_1$ — формула разметки после первого применения правила прямого прослеживания

2. свёртка программы — преобразование частично или полностью размеченного ИГР в тройку Хоара,

3. эквивалентное преобразование — изменение графа по правилам эквивалентных преобразований языка Пифагор,

4. расщепление — получение двух и более ИГР из одного исходного.

Весь процесс доказательства на ИГР можно представить в виде дерева, корень которого — исходный ИГР, дочерние узлы получаются из родительских выполнением одного из преобразований графа, а листья — полностью размеченные ИГР, над которыми проводится полная свёртка и преобразование в формулы.

Таким образом, формализация семантики и определение аксиоматической системы для языка спецификации использовано для построения аксиоматической теории на базе исчисления Хоара, позволяющей доказывать корректность программ на языке Пифагор; предложен наглядный способ проведения доказательства непосредственно на информационном графе программы.

В **третьей главе** предложен метод доказательства корректности рекурсивных ФПП программ с использованием построенной аксиоматической теории для языка Пифагор. Процесс доказательства разделяется на два этапа: доказательство частичной корректности (в предположении, что программа завершается) и доказательство завершения программы.

Для сведения задачи к доказательству корректности одной функции, предложен алгоритм преобразования произвольной рекурсивной функции в прямую.

Доказательство частичной корректности проводится по правилам (1) и (2). При этом рекурсивный вызов рассматривается так же, как вызовы других функций, а в качестве теоремы в правиле (1) используется доказываемая тройка Хоара, у которой текущий аргумент заменяется на аргумент рекурсивного вызова, а идентификатор результата — на идентификатор выходной дуги рекурсивного вызова. Данное доказательство основано на индукции. Базу индукции даёт доказательство корректности всех ветвей функции, не содержащих рекурсию; предположением индукции является корректность доказываемой тройки для всех рекурсивных вызовов.

При доказательстве завершения рекурсивной функции используется трансфинитная индукция. Задаётся фундированное множество S с отношением порядка $\prec$ и ограничивающая функция φ, принимающая значения из фундированного множества, если её аргумент удовлетворяет предусловию рассматриваемой функции. Необходимо показать, что значение ограничивающей функции для аргумента текущего вызова функции будет больше, чем для аргументов каждого рекурсивного вызова. Аргументы ограничивающей функции совпадают с аргументами рекурсивной функции, поэтому условия завершения функции могут быть добавлены в пред- и постусловие рекурсивной функции, и тогда доказательство истинности тройки Хоара станет доказательством тотальной корректности. Если фундированное множество S с отношением порядка $\prec$ и ограничивающая функция φ введены корректно, то для рекурсивной функции `Prog` с условием частичной корректности $\boxed{P(x)}$ `Prog(x)` $\boxed{Q}$, условие тотальной корректности будет следующим:

$$\boxed{P(x)}\ \texttt{Prog(x)}\ \boxed{Q \bigwedge_i (\varphi(x) < \varphi(x_i))},$$

где x_i — идентификатор аргумента i-го рекурсивного вызова, также являющийся идентификатором входной дуги оператора интерпретации, выполняющего рекурсивный вызов функции.

Программа может состоять из нескольких функций, корректность каждой доказывается отдельно. Если функции связаны косвенной рекурсией, то проводятся преобразования с получением прямой рекурсии. Основным способом сведения произвольной рекурсии к прямой является построение универсальной рекурсивной функции. В простых случаях, косвенную рекурсию можно свести к прямой рекурсии с помощью объединения тел связанных функций в одну функцию. Алгоритм преобразования произвольной рекурсивной функции A в прямую рекурсию состоит в следующем:

1. строится граф всех связанных функций $G(A)$ — это ориентированный граф, вершины которого помечены именами функций, которые вызываются из A, как непосредственно в её теле, так и через цепочку вызовов других функций, а дуги графа направлены из вызывающей функции в вызываемую;

2. из графа удаляются все нерекурсивные функции, которым будут соответствовать вершины, не входящие ни в один цикл (их корректность доказывается отдельно);

3. из графа удаляются все рекурсивные функции, не связанные с A, сюда относятся все узлы для которых любой цикл, проходящий через этот узел, не проходит через вершину A (их корректность доказывается отдельно);

4. узлы с косвенно рекурсивными функциями имеют только одну входную дугу, их код может быть объединён с кодом функции из узла, смежного по входной дуге рассматриваемого узла;

5. для всех оставшихся функций графа задаётся универсальная рекурсивная функция.

Показано, что объединение кода функций и построение универсальной рекурсивной функции не изменяет корректность исходной программы.

Таким образом, разработанный алгоритм позволяет доказывать корректность произвольной рекурсивной функции, которая предварительно преобразуется в прямую рекурсию путём построения универсальной рекурсивной функции и объединения кода связанных функций. Сам процесс доказательства состоит в установлении частичной корректности функции и доказательстве завершения программы.

В **четвёртой главе** представлена архитектура системы, обеспечивающей поддержку доказательства корректности ФПП программ на языке Пифагор. Описан прототип системы, разработанный на базе данной архитектуры.

На рисунке 2 приведена общая схема системы. В системе можно выделить несколько элементов: модуль доказательства корректности программы, модуль управления библиотекой аксиом и теорем, модуль доказательства формул (верификатор формул). Модуль доказательства формул обособлен от остальной части системы, так как является сторонним средством и может не использоваться при доказательстве, в этом случае все его действия выполняются пользователем.

Принцип работы системы состоит в следующем. Пользователь передаёт системе программу на языке Пифагор и спецификацию программы на языке спецификации. Модуль доказательства корректности программы формирует исходный ИГР и начинает процесс доказательства, заключающийся в последовательности преобразований ИГР и формировании дерева доказательства. Пользователь управляет процессом, выбирая тип следующего преобразования. Эквивалентные преобразования и разметка готовых операторов, отличных от оператора интерпретации, проводятся полностью автоматически. Для разметки выходных дуг операторов интерпретации используется информация об аксиомах и уже доказанных теоремах из библиотеки аксиом и теорем. Модуль доказательства корректности программы посылает запросы к модулю управ-

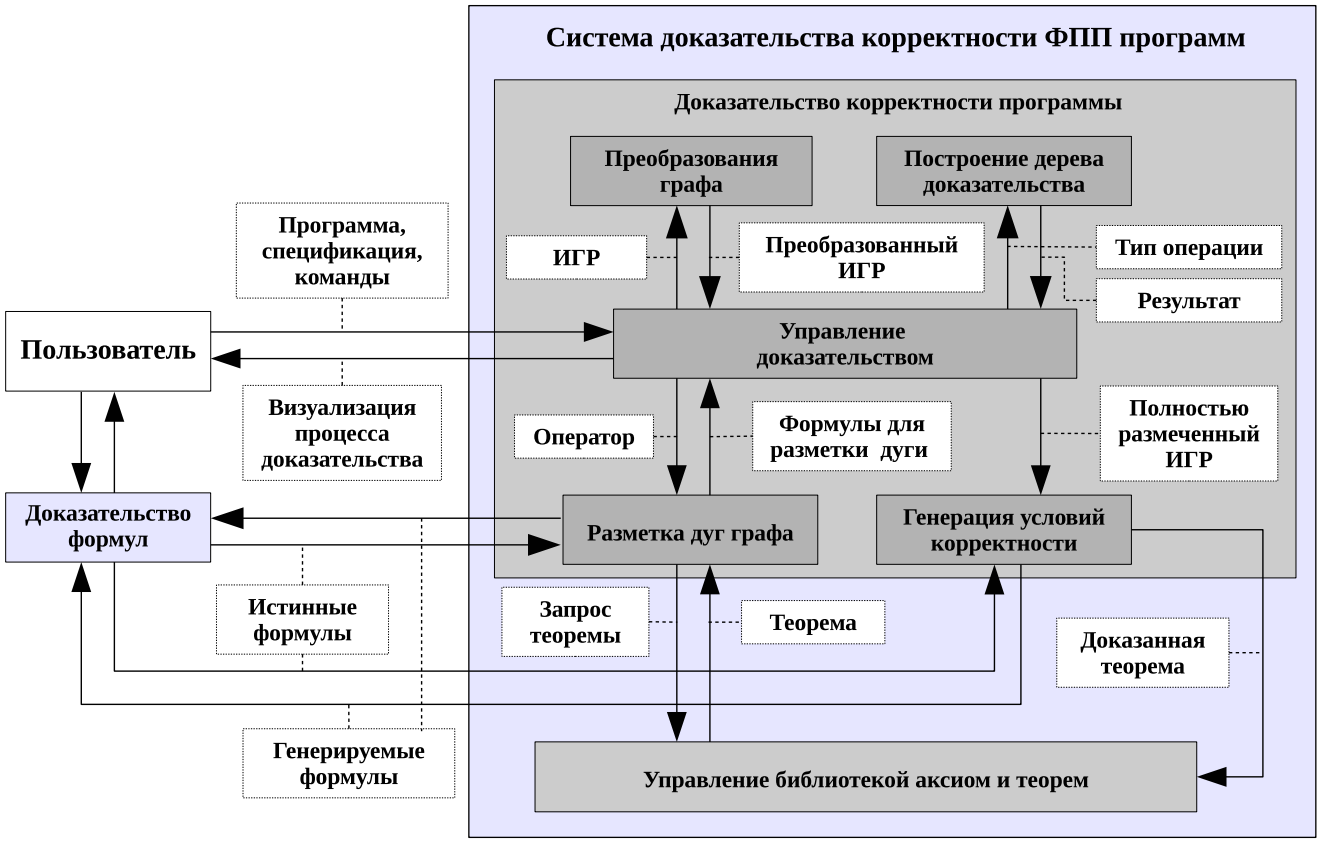

Рисунок 2 — Обобщённая структура системы поддержки формальной верификации; модули обозначены закрашенными прямоугольниками, стрелки — информационные связи между модулями

ления библиотекой аксиом и теорем. В случае, если запрашиваемая функция отсутствует в библиотеке, выдается ошибка о невозможности разметки. Если аксиомы (теоремы) для рассматриваемой функции присутствуют в библиотеке, то модуль доказательства корректности программы проводит их отбор. Для каждой аксиомы (теоремы) из набора аксиом (теорем) для рассматриваемой функции формируется условие применимости данной аксиомы (теоремы) на языке спецификации. Это условие передаётся стороннему модулю доказательства формул, и, если оно истинно или выполнимо, то аксиома (теорема) используется для разметки, иначе она отбрасывается. После того как все дуги в информационном графе программы размечены, проводится полная свёртка. Тройки Хоара, соответствующие полученным ИГР, преобразуются в формулы, которые передаются верификатору для проверки истинности. Если все формулы истинны, то программа корректна, а ее исходная тройка Хоара — теорема. Модуль доказательства корректности программы имеет возможность передать полученную теорему модулю управления библиотекой аксиом и теорем для сохранения в библиотеке.

Реализованный прототип системы включает в себя:

1. модуль поддержки РИГ (РИГ — реверсивный информационный граф,

описывающий зависимости по данным), модифицированный автором и интегрированный в систему;

2. модуль поддержки термов, содержащий транслятор термов на языке спецификации из текстового представления во внутреннее;

3. модуль ИГР, отвечающий за внутренне представление ИГР и осуществляющий основные операции по созданию и преобразованию графа, автоматизированной разметке дуг графа;

4. модуль поддержки дерева доказательства, позволяющий сохранять последовательность шагов при доказательстве корректности программы;

5. библиотеку аксиом и теорем, используемую для разметки выходных дуг операторов интерпретации;

6. графическую оболочку, предназначенную для взаимодействия с пользователем и осуществляющую управление взаимодействием между остальными модулями системы.

Система визуализирует процесс доказательства. Часть этапов доказательства выполняется автоматически; для другой части этапов, проходящих в интерактивном режиме, система предоставляет всевозможные варианты пути доказательства, из которых пользователь выбирает допустимые.

Таким образом, для облегчения процесса формальной верификации программ на языке Пифагор разработана архитектура системы поддержки процесса доказательства, а также реализован прототип данной системы.

В **заключении** приведены выводы работы и сформулированы основные результаты.

Основные результаты и выводы

Работа посвящена повышению надёжности и корректности функционально-потоковых параллельных программ посредством разработки для них методов формальной верификации. Полученные результаты имеют значение для развития теоретических основ программирования, включающих языки программирования, технологии программирования, автоматизацию программирования, теоретические основы системного и прикладного программного обеспечения.

Основные научные и практические результаты.

1. В результате исследования применимости существующих методов формальной верификации к доказательству корректности ФПП программ, разработан метод верификации ФПП программ на языке Пифагор, основанный на исчислении Хоара, и по сложности сравнимый с методами доказательства корректности для последовательных программ. Для разработки метода потребовалось формализовать семантику языка Пифагор, разработать язык спецификации свойств программ и создать аксиоматическую теорию на базе исчисления Хоара.

2. Предложен метод доказательства завершения программ на языке Пифагор. Метод позволяет изменить спецификацию программы таким образом, что доказательство частичной корректности на базе исчисления Хоара одновременно доказывает и завершение программы.

3. Предложен метод удаления взаимной рекурсии нескольких функций. Метод позволяет преобразовывать любую функцию ФПП программы в функцию с прямой рекурсией. Это упрощает процесс доказательства тем, что позволяет работать только с одной функцией при доказательстве корректности методом, основанным на исчислении Хоара.

4. Разработана архитектура инструментального средства поддержки доказательства корректности ФПП программ. Разработан прототип инструментального средства, позволяющий доказывать корректность программ на языке Пифагор. Данная система позволяет упростить доказательство за счёт визуализации процесса и автоматизации построения дерева доказательства.

Перспективы развития направления исследований. Полученные результаты позволяют проводить формальную верификацию ФПП программ. Из верифицированных программ в перспективе можно сформировать библиотеку неограниченно параллельных программ и, при необходимости, преобразовывать в программы для различных реальных архитектур, которые также будут корректны, если доказать корректность правил преобразования.

Публикации по теме диссертации

В изданиях, рекомендованных ВАК:

1. Легалов, А.И. Преобразование хвостовых рекурсий в функционально-потоковых параллельных программах / А.И. Легалов, О.В. Непомнящий, И.В. Матковский, М.С. Кропачева (Ушакова) // Моделирование и анализ информационных систем. — 2012. — Т. 19, № 5. — С. 48–58.

2. Кропачева (Ушакова), М.С. Формальная верификация программ, написанных на функционально-потоковом языке параллельного программирования / М.С. Кропачева (Ушакова), А.И. Легалов // Моделирование и анализ информационных систем. — 2012. — Т. 19, № 5. — С. 81–99.

3. Ushakova, M.S. Automation of Formal Verification of Programs in the Pifagor Language / M.S. Ushakova, A.I. Legalov // Modeling and Analysis of Information Systems. — 2015. — Vol. 22, N. 4. — P. 578–589.

4. Легалов, А.И. Инструментальная поддержка создания и трансформации функционально-потоковых параллельных программ / А.И. Легалов, В.С. Васильев, И.В. Матковский, М.С. Ушакова // Труды Института системного программирования РАН. — 2017. — Т. 29, № 5. — С. 165–184.

5. Ушакова, М.С. Верификация программ со взаимной рекурсией на языке Пифагор / М.С. Ушакова, А.И. Легалов // Моделирование и анализ информа-

ционных систем. — 2018. — Т. 25, № 4. — С. 358–381.

6. Легалов, А.И. Динамически изменяющийся параллелизм с асинхронно-последовательными потоками данных / А.И. Легалов, И.В. Матковский, М.С. Ушакова, Д.С. Романова // Моделирование и анализ информационных систем. — 2020. — Т. 27, № 2. — С. 164–179.

В индексируемых базах данных:

7. Kropacheva (Ushakova), M.S. Formal Verification of Programs in the Pifagor Language / M.S. Kropacheva (Ushakova), A.I. Legalov // Parallel Computing Technologies (PaCT-2013) 12th International Conference, September 30 – October 4, 2013. Saint-Petersburg, Russia. LNCS. — 2013. — Vol. 7979. — P. 80–89.

8. Kropacheva (Ushakova), M.S. Formal Verification of Programs in the Functional Data-Flow Parallel Language [перевод] / M.S. Kropacheva (Ushakova), A.I. Legalov // Automatic Control and Computer Sciences. — 2013. — Vol. 47, N. 7. — P. 373–384.

9. Legalov, A.I. Tail Recursion Transformation in Functional Dataflow Parallel Programs [перевод] / A.I. Legalov, O.V. Nepomnyaschy, I.V. Matkovsky, M.S. Kropacheva (Ushakova) // Automatic Control and Computer Sciences. — 2013. — Vol. 47, N. 7. — P. 366–372.

10. Legalov, A.I. A Toolkit For The Development Of Data-Driven Functional Parallel Programmes / A.I. Legalov, V.S. Vasilyev, I.V. Matkovskii, M.S. Ushakova // Communications in Computer and Information Science. — 2018. — Vol. 910. — P. 16–30.

11. Ushakova, M.S. Verification Of Programs With Mutual Recursion In Pifagor Language [перевод] / M.S. Ushakova, A.I. Legalov // Automatic Control and Computer Sciences. — 2018. — Vol. 52, N. 7. — P. 850–866.

12. Legalov, A.I. Dynamically Changing Parallelism with Asynchronous Sequential Data Flows / A.I. Legalov, I.V. Matkovskii, M.S. Ushakova, D.S. Romanova // Automatic Control and Computer Sciences. — 2021. — Vol. 55, N. 7. — P. 636–646.

Другие издания:

13. Удалова, Ю.В. Об отладке и верификации программ на функционально-потоковом параллельном языке Пифагор / Ю.В. Удалова, М.С. Кропачева (Ушакова) // Материалы V всероссийской научно-технической конференции «Молодёжь и наука: начало XXI века». — Красноярск, 2009. — С. 40–43.

14. Кропачева (Ушакова), М.С. Верификация функционально-потоковых параллельных программ методом поиска слабейшего предусловия / М.С. Кропачева (Ушакова) // Материалы XI Всероссийской научно-практической конференции «Проблемы информатизации региона» (ПИР-2009). — Красноярск: РИЦ СибГТУ, 2009. — С. 136–139.

15. Кропачева (Ушакова), М.С. Формализация семантики функционально-

потокового языка параллельного программирования Пифагор / М.С. Кропачева (Ушакова) // Материалы XII Всероссийской научно-практической конференции «Проблемы информатизации региона» (ПИР-2011). — Красноярск: Сибирский федеральный университет, 2011. — С. 144–148.

16. Кропачева (Ушакова), М.С. Формальная верификация параллельных программ / М.С. Кропачева (Ушакова) // Труды XII Международной научной конференции «Интеллект и наука». — Красноярск: Центр информатизации, 2012. — С. 130–131.

17. Кропачева (Ушакова), М.С. Система автоматизированного доказательства корректности функционально-потоковых параллельных программ / М.С. Кропачева (Ушакова) // Молодёжь и наука: сборник материалов VIII Всероссийской научно-технической конференции. — Красноярск: Сиб. федер. ун-т., 2012. — С. 62–65.

18. Кропачева (Ушакова), М.С. Формальная верификация параллельных программ / М.С. Кропачева (Ушакова) // Исследования наукограда. — 2012. — № 2. — С. 35–38.

19. Кропачева (Ушакова), М.С. Аксиоматический подход к формальной верификации рекурсивных программ на функционально-потоковом языке параллельного программирования / М.С. Кропачева (Ушакова) // Параллельные вычислительные технологии (ПаВТ'2013): труды международной научной конференции (г. Челябинск, 1–5 апреля 2013 г.). — Челябинск: Издательский центр ЮУрГУ, 2013. —С. 421–431.

20. Кропачева (Ушакова), М.С. Общая концепция и архитектура программного средства инструментальной поддержки методов формальной верификации функционально-потоковых параллельных программ / М.С. Кропачева (Ушакова) // Многоядерные процессоры, параллельное программирование, ПЛИС, системы обработки сигналов: сб. ст. — Барнаул: Барнаул, 2013. — С. 113–116.

21. Ушакова, М.С. Инструментальная поддержка формальной верификации программ, написанных на языке функционально-потокового параллельного программирования / М.С. Ушакова, А.И. Легалов // Вестник Южно-Уральского государственного университета. Серия Вычислительная математика и информатика. — 2015. — Т. 4, № 2. — С. 58–68.

22. Ушакова, М.С. Семантика типов данных функционально-потокового языка параллельного программирования Пифагор / М.С. Ушакова // Образовательные ресурсы и технологии. — 2016. — № 2 (14). — С. 263–269.

23. Удалова, Ю.В. Верификация и доказательство завершения функционально-потоковых параллельных программ / Ю.В. Удалова, М.С. Ушакова // Языки программирования и компиляторы — 2017: труды конференции / Южный федеральный университет; под ред. Д.В. Дуброва. — Ростов-на-Дону: Изд-

во ЮФУ, 2017. — С. 248–251.

24. Легалов, А.И. Особенности разработки и преобразования функционально-потоковых параллельных программ / А.И. Легалов, М.С. Ушакова // Суперкомпьютерные дни в России: труды международной конференции (Москва, 24-25 сентября 2018 г.). Суперкомпьютерный консорциум университетов России, Российская академия наук. — М.: Московский государственный университет имени М.В. Ломоносова, 2018. — С. 999–1000.

25. Ушакова, М.С. Использование SMT-решателей для доказательства условий корректности программ на языке Пифагор / М.С. Ушакова // Проспект Свободный — 2018: материалы Международной студенческой конференции (Красноярск, 23-27 апреля 2018 г.). — Красноярск: СФУ, 2018. — С. 847–850.

Свидетельства о государственной регистрации программ для ЭВМ:

26. Легалов, А.И. Свидетельство о государственной регистрации программы для ЭВМ № 2013611434 Российская Федерация. Синтаксический анализатор текстового представления реверсивного информационного графа / А.И. Легалов, О.В. Непомнящий, И.В. Матковский, М.С. Кропачева (Ушакова); заявитель и правообладатель Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Сибирский федеральный университет» (СФУ). — опубл. 09.01.2013. — 1 с.

27. Ушакова, М.С. Свидетельство о государственной регистрации программы для ЭВМ № 2021663755 Российская Федерация. Инструментальное средство для формальной верификации функционально-потоковых параллельных программ на языке Пифагор на основе исчисления Хоара / Ушакова М.С.; заявитель и правообладатель Федеральное государственное автономное образовательное учреждение высшего образования «Сибирский федеральный университет» (СФУ). — № 2021662716; заявл. 09.08.2021; опубл. 23.08.2021. — 1 с.