Московский государственный университет
имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики

На правах рукописи

Коннов Игорь Владимирович

ВЕРИФИКАЦИЯ ПАРАМЕТРИЗОВАННЫХ МОДЕЛЕЙ РАСПРЕДЕЛЁННЫХ СИСТЕМ

Специальность 05.13.11 — математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата физико-математических наук

МОСКВА
2008

Работа выполнена на кафедре автоматизации систем вычислительных комплексов факультета вычислительной математики и кибернетики Московского государственного университета имени М.В. Ломоносова.

Научные руководители:	доктор физико-математических наук, профессор, академик РАЕН Смелянский Руслан Леонидович; кандидат физико-математических наук, доцент Захаров Владимир Анатольевич.
Официальные оппоненты:	доктор физико-математических наук, профессор Соколов Валерий Анатольевич; кандидат физико-математических наук, старший научный сотрудник Кулямин Виктор Вячеславович.
Ведущая организация:	Вычислительный центр имени А.А. Дородницына Российской академии наук.

Защита состоится «28» ноября 2008 г. в 11:00 на заседании диссертационного совета Д 501.001.44 при Московском государственном университете имени М.В. Ломоносова по адресу: 119991, ГСП-1, Москва, Ленинские горы, МГУ, 2-ой учебный корпус, факультет вычислительной математики и кибернетики, аудитория 685.

С диссертацией можно ознакомиться в библиотеке факультета ВМиК МГУ имени М.В. Ломоносова, с текстом автореферата — на официальном сайте ВМиК МГУ имени М.В. Ломоносова: `http://www.cs.msu.su` в разделе «Наука» — «Работа диссертационных советов» — «Д 501.001.44».

Автореферат разослан «_____» октября 2008 г.

Ученый секретарь
диссертационного совета Д 501.001.44
профессор Н.П. Трифонов

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность работы. Одним из наиболее распространенных подходов к проверке правильности программ является метод верификации моделей программ (Model Checking, MC). Суть метода такова. Для заданной программы строится её логическая модель, множество трасс которой покрывает множество вычислений программы. Спецификация программы, описывающая правильное поведение программы, задаётся в виде логических формул. Доказательство того, что программа удовлетворяет спецификации, заключается в проверке выполнимости указанных логических формул на модели программы. Чаще всего в качестве моделей программ выбираются размеченные системы переходов с конечным множеством состояний, а спецификации программ задаются в виде формул одной из темпоральных логик. Задача проверки выполнимости темпоральной формулы на модели с конечным множеством состояний алгоритмически разрешима. Более того, для многих темпоральных логик построены эффективные алгоритмы проверки выполнимости спецификации за время, линейное относительно размера модели[1]. Благодаря этому метод MC получил широкое распространение как практически пригодный автоматический метод верификации программ со сложным поведением.

В настоящее время существует несколько десятков систем верификации программ, разработанных на основе метода MC. В этих системах используются различные модели программ, темпоральные логики и алгоритмы проверки выполнимости логических формул. Наиболее известными из них являются системы верификации Spin, SMV, RuleBase, Java Pathfinder, SLAM, BLAST, которые успешно эксплуатируются во многих крупнейших компаниях, занимающихся разработкой вычислительной техники и программного обеспече-

[1]Кларк Э., Грамберг О., Пелед Д. Верификация моделей программ: Model Checking. — М.: Издательство Московского центра непрерывного математического образования, 2002.

ния (Microsoft, Intel, NASA, IBM и др.). При помощи этих инструментальных средств удается проводить проверку правильности логических описаний микроэлектронных схем, драйверов операционных систем, сетевых протоколов, распределенных алгоритмов. Известно немало примеров обнаружения при помощи этих средств изощренных и трудноуловимых программных ошибок.

Вместе с тем применение метода МС для верификации некоторых сложных программ сталкивается с принципиальными трудностями. Подавляющее большинство алгоритмов МС позволяют эффективно верифицировать только модели программ с конечным числом состояний. Однако существуют и такие распределённые программные системы, для описания которых приходится использовать бесконечные семейства моделей со сколь угодно большим числом состояний. К числу таких систем относятся: распределённые алгоритмы (волновые алгоритмы, распределения ресурсов, взаимного исключения доступа к критической секции, избрания лидера, обнаружения завершения, согласованного принятия транзакции), сетевые протоколы (кольцевые сети с маркерами, протоколы маршрутизации, протоколы обеспечения качества сервиса), аппаратные схемы (управления доступом к шине при различном числе клиентских устройств, обеспечения когерентности кэшей).

Модели распределённых систем такого вида состоят из однотипных взаимодействующих процессов; число процессов является параметром, зависящим от начальной конфигурации, и может быть сколь угодно большим. Вследствие этого, обычные алгоритмы решения задачи МС для конечных моделей программ не могут гарантировать корректной проверки параметризованных моделей программ. Таким образом, возникает задача верификации параметризованных моделей распределенных систем (Parameterized Model Checking, PMC), для решения которой нужны специальные алгоритмы.

Настоящая диссертация посвящена исследованию и решению задачи PMC для одного класса параметризованных моделей распределенных программ.

Цель работы. Цель диссертационной работы — разработка и анализ алгоритмов верификации параметризованных моделей распределённых программ с произвольным числом однотипных асинхронно-взаимодействующих процессов. При этом новые разработанные алгоритмы решения задачи PMC должны быть совместимы с теми способами представления конечных моделей программ и алгоритмами решения задачи MC для конечных моделей, которые используются в современных инструментальных системах верификации программ. Благодаря этому указанные системы верификации конечных моделей программ могут быть легко адаптированы для проверки правильности гораздо более широких классов распределенных программ.

Методы исследования. При получении основных результатов работы диссертации использовались методы математической логики, алгебры, теории графов, теории автоматов и теории формальных языков.

Основные результаты работы.

1. Предложен новый метод верификации систем асинхронно-взаимодействующих процессов, позволяющий автоматически вычислять конечные инварианты параметризованных моделей программ и сводить задачу верификации бесконечных параметризованных моделей программ к задаче верификации конечных моделей.

2. На основе предложенного метода разработаны новые алгоритмы верификации параметризованных систем асинхронно-взаимодействующих процессов, сочетающие методы верификации моделей программ и проверки отношений симуляции между моделями программ.

3. На основе предложенного метода и алгоритмов разработана и реализована экспериментальная система автоматической верификации программ, с помощью которой доказана корректность поведения ряда рас-

пределенных алгоритмов и сетевых протоколов, применяющихся на практике.

Научная новизна. В диссертации представлены новые ослабленные отношения симуляции на множестве размеченных систем переходов, позволяющие проводить эффективный поиск конечных инвариантов асинхронных параметризованных моделей программ. На основе разработанного алгоритма вычисления полублочной симуляции впервые предложен общий метод вычисления инвариантов для параметризованных моделей систем асинхронно-взаимодействующих процессов, не требующий применения абстракции. Для обоснования корректности предложенного метода была также введена и использована новая разновидность отношения симуляции моделей — квазиблочная симуляция.

Практическая ценность. Новые методы и алгоритмы вычисления инвариантов, разработанные в настоящей диссертационной работе, позволяют сводить задачу верификации бесконечных семейств моделей программ с асинхронным параллелизмом, параметризованных по числу однотипных процессов, к хорошо изученной задаче верификации конечных моделей, для решения которой существуют многочисленные инструментальные средства.

На основе разработанных алгоритмов спроектирована и реализована экспериментальная система верификации параметризованных моделей распределённых систем. Эта экспериментальная система была интегрирована с одним из наиболее распространенных инструментальных средств верификации конечных моделей программ SPIN и успешно опробована на нескольких примерах параметризованных моделей распределённых алгоритмов и сетевых протоколов. Результаты проведенных экспериментов продемонстрировали, что новые алгоритмы вычисления инвариантов параметризованных моделей распределенных программ существенно расширяют область применения ма-

тематических методов проверки правильности программ и построенных на их основе систем автоматической верификации программ.

Апробация работы. Результаты, представленные в работе, докладывались на объединённом научно-исследовательском семинаре кафедр автоматизации систем вычислительных комплексов, системного программирования и алгоритмических языков факультета ВМиК МГУ имени М.В. Ломоносова под руководством профессора М.Р. Шура-Бура, на научных семинарах лаборатории вычислительных комплексов кафедры автоматизации систем вычислительных комплексов факультета ВМиК МГУ имени М.В. Ломоносова под руководством профессора Р.Л. Смелянского, рабочих совещаниях группы проекта INTAS 05-1000008-8144 «Practical Formal Verification Using Automated Reasoning and Model Checking», а также на следующих конференциях:

- Всероссийские конференции «Методы и средства обработки информации» (Москва, октябрь 2003 и 2005 гг.);
- Научная конференция «Тихоновские чтения» (Москва, октябрь 2005 г.);
- Седьмая международная конференция «Дискретные модели в теории управляющих систем» (Москва, март 2006 г.);
- Научная конференция «Ломоносовские чтения» (Москва, апрель 2007);
- Научный семинар «Методы порождения инвариантов программ» (Workshop on Invariant Generation) (Австрия, Хагенберг, июнь 2007 г.);
- Пятая Всероссийская научная конференция студентов, аспирантов и молодых учёных «Технологии Microsoft в теории и практике программирования», секция «Теоретическое программирование» (Москва, апрель 2008 г.).

Работа была выполнена при поддержке грантов INTAS и РФФИ.

Публикации. По теме диссертации имеется 7 публикаций, список которых приводится в конце автореферата.

Структура и объем диссертации. Диссертация состоит из введения, шести глав, списка литературы и двух приложений. Объём работы — 142 страницы, с приложениями — 198 страниц. Список литературы содержит 110 наименований.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Работа состоит из введения, шести глав, заключения и двух приложений.

Во **введении** обоснована актуальность диссертационной работы, сформулирована цель исследований и представлены выносимые на защиту научные положения.

В **первой главе** проводится и обосновывается выбор математической модели распределенных программных систем, а также формулируется общая задачи верификации параметризованных моделей систем взаимодействующих процессов (Parameterized Model Checking, PMC).

В начале главы рассмотрены примеры широко известных распределённых систем. Анализ этих примеров позволяет сформулировать основные допущения и требования, которые учитываются при построении параметризованных моделей распределенных систем, а именно:

- отсутствие глобальных часов,
- дискретное изменение состояний процессов,
- асинхронное выполнение процессов системы,
- взаимодействие процессов посредством синхронного обмена сообщениями,

- абсолютная надежность процессов и каналов связи.

Для описания топологии коммуникационной сети распределенной системы используются сетевые грамматики.

В качестве моделей распределённых программ используются *размеченные системы переходов (LTS)*, состоящие из конечного множества размеченных состояний, множества начальных состояний и отношения размеченных переходов между состояниями. На множестве систем переходов определяется операция параллельной композиции. Отдельные процессы распределенной системы моделируются при помощи LTS, которые описываются в явном виде; модель всей распределенной системы образуется в результате параллельной композиции LTS процессов: $M = P_1 \parallel \cdots \parallel P_n$.

В качестве языка спецификаций поведения распределённых систем выбираются темпоральные логики: линейного времени LTL, ветвящегося времени CTL и обобщение обеих логик — логика $\mathsf{CTL}^\star$.

Задача верификации параметризованных моделей распределённых систем (PMCP1), рассматриваемая в данной работе, формулируется следующим образом. Пусть заданы

1. бесконечное семейство конечных моделей $\mathcal{F} = \{M_n\}_{n=1}^{\infty}$, параметризованное по параметру $n \in \mathbb{N}$; каждая модель $M_n = Q \parallel P_1 \parallel \cdots \parallel P_n$ состоит из LTS фиксированного процесса Q и n экземпляров процесса-прототипа P — LTS процессов P_i;

2. конечное множество $P_i,\ i \in I,\ I \subseteq \mathbb{N}$, *наблюдаемых процессов*, каждый из которых является некоторым вариантом процесса-прототипа P;

3. спецификация, представленная формулой темпоральной логики φ, зависящей от переменных выделенных процессов P_i, $i \in I$, и переменных процесса Q.

Требуется проверить выполнимость формулы φ на всех моделях семейства $\mathcal{F}$, т. е. убедиться в том, что соотношение $M_n \models \varphi$ соблюдается для всех n, $n \geq 1$.

В конце главы проводится обсуждение некоторых других вариантов задачи PMC.

Во **второй главе** представлен обзор и сравнительный анализ известных подходов к решению задачи PMC для различных параметризованных моделей распределенных систем. Принимая во внимание характерные особенности выбранной модели распределенных систем, предпочтение было отдано методам, основанным на поиске инвариантов параметризованной модели.

Известно, что в общем случае задача PMC алгоритмически неразрешима[2]. Однако существуют классы параметризованных моделей, для которых задача PMCP может быть решена. Обзор методов верификации параметризованных моделей проводится относительно следующих критериев:

- область применимости метода (топология параметризованных моделей, виды параллелизма, способы взаимодействия процессов, ограничения на структуру LTS процессов),
- виды проверяемых спецификаций,
- возможность полной алгоритмизации.

Основными методами решения задачи PMC являются аналитические методы редукции, методы абстракции, символьные методы и методы поиска инвариантов. Показано, что первые три указанных метода не позволяют достичь поставленной цели: аналитический метод редукции применим к ограниченному классу моделей, метод абстракции требует участия эксперта, а символьные методы требуют повторного описания модели на специальном языке и участия эксперта для гарантии завершаемости.

[2]Apt K., Kozen D. Limits for automatic verification of finite-state concurrent systems. *Information Processing Letter*, vol. 15, Pp. 307– 309, 1986.

Методы поиска инвариантов осуществляют сведение задачи PMC к задаче MC для конечных моделей за счет конструирования для заданной параметризованной модели $\mathcal{F} = \{M_n\}_{n=1}^{\infty}$ такой конечной модели Inv (*инвариант*), которая обладает следующими двумя свойствами:

1. модель Inv подобна некоторой модели M_k семейства $\mathcal{F}$,

2. параллельная композиция $Inv \parallel P$ подобна модели Inv.

Если отношение подобия моделей обладает определенным набором свойств (монотонность параллельной композиции, консервативность отношения выполнимости для темпоральных формул), то верификация параметризованной модели $\mathcal{F}$ сводится к решению задачи MC для конечных моделей $M_1, M_2, \ldots, M_{k-1}, Inv$. Отношения подобия на множестве моделей, а также способы конструирования модели-инварианта Inv могут быть разнообразны, и этим обеспечивается возможность широкого применения метода поиска инвариантов для решения задачи PMC. Отношение подобия *консервативно* для класса спецификаций Φ, если для любых подобных моделей M_1 и M_2 и любой спецификации $\varphi \in \Phi$ из выполнимости $M_2 \models \varphi$ следует выполнимость $M_1 \models \varphi$. Отношение *монотонно* для операции параллельной композиции, если из подобия пар M_1 и M_2, M_3 и M_4 следует подобие $M_1 \parallel M_3$ и $M_2 \parallel M_4$.

В *разделе 2.5* обосновывается выбор метода поиска инвариантов в качестве основного подхода к решению того варианта задачи PMC, который рассматривается в диссертационной работе. Предпочтение этому методу отдается по следующим причинам:

- возможность полной и эффективной алгоритмизации,

- применимость метода для верификации параметризованных моделей с разнообразной топологией коммуникационной сети,

- сочетаемость метода с алгоритмами и инструментальными средствами верификации конечных моделей.

Метод поиска инвариантов в сочетании с другими методами верификации успешно использовался для верификации синхронных систем взаимодействующих процессов[3]. Одна из целей исследований настоящей диссертационной работы — распространить методы поиска инвариантов для верификации параметризованных моделей асинхронных систем взаимодействующих процессов.

В **третьей главе** описываются наиболее важные базовые понятия, используемые при построении моделей программ и их верификации, а именно размеченные системы переходов (LTS), темпоральные логики, сетевые грамматики. В терминах введённых понятий формулируется математическая задача проверки выполнимости спецификации, заданной формулой темпоральной логики, на параметризованных моделях программ, представленных посредством размеченных систем переходов и сетевых грамматик.

Модели распределённой системы описываются в виде LTS. В *разделе 3.1* LTS определяется в виде шестерки $\langle S, S^0, A, R, \Sigma, L\rangle$, в которой S — конечное множество состояний, S^0 — подмножество S начальных состояний системы, A — конечное множество действий, $R \subseteq S \times A \times S$ — отношение размеченных переходов, Σ — конечное множество булевых переменных системы, L — функция разметки состояний $L : S \to 2^{\Sigma}$. Конечная или бесконечная последовательность $s_1 \stackrel{a_1}{\to} s_2 \stackrel{a_2}{\to} \ldots \stackrel{a_{n-1}}{\to} s_n \stackrel{a_n}{\to} \ldots$ переходов LTS называется путём в LTS, если $(s_i, a_i, s_{i+1}) \in R$ для всех $i,\ i \geq 1$. Путь в модели — это абстрактный образ вычисления моделируемой распределённой системы.

В *разделе 3.2* определяется операция *асинхронной параллельной композиции* $M_i \parallel_\Gamma M_2$ для LTS $M_i = \langle S_i, S_i^0, A_i, R_i, \Sigma_i, L_i\rangle$, $i = 1, 2$. Одна из особен-

[3]Clarke E., Grumberg O., Jha S. Verifying parameterized networks using abstraction and regular languages. *In Proc. of 6-th International Conference on Concurrency Theory*, Pp. 395–407, 1995.

ностей этой операции состоит в том, что она требует явного указания правил синхронного взаимодействия моделей M_1 и M_2. Правила взаимодействия задаёт *синхронизатор* $\Gamma = (\Delta, ^-)$, в котором множество $\Delta \subseteq A_1$ описывает множество синхронизируемых действий модели M_1, а функция $^- : \Delta \to A_2$ описывает соответствующие действия модели M_2, синхронно выполняющиеся с действиями из множества Δ.

Раздел 3.3 посвящен сетевым грамматикам, с помощью которых задается описание топологии и порождаются модели параметризованного семейства. Контекстно-свободной сетевой грамматикой называется такая четвёрка $G = \langle \mathcal{T}, \mathcal{N}, \mathcal{P}, \mathcal{S} \rangle$, представленная конечным множеством терминалов $\mathcal{T}$ (в роли терминалов вступают LTS), конечным множеством нетерминалов $\mathcal{N}$, стартовым нетерминалом $\mathcal{S}$ и конечным множеством $\mathcal{P}$ правил вывода, имеющих вид $X \to Y_1[\mathcal{R}_1] \parallel_{\Gamma_1} \cdots \parallel_{\Gamma_{n-1}} Y_n[\mathcal{R}_n]$. В каждом таком правиле участвуют нетерминал $X \in \mathcal{N}$, терминалы или нетерминалы $Y_1, \ldots, Y_n \in \mathcal{N} \cup \mathcal{T}$, синхронизаторы действий процессов Γ_i, $1 \leq i \leq n-1$, а также функции переименования действий $\mathcal{R}_i$. С помощью сетевой грамматики порождаются LTS, которые и образуют бесконечные параметризованные семейства конечных моделей программ. Аналогично выводу в контекстно-свободных грамматиках для каждого символа X из множества $\mathcal{N} \cup \mathcal{T}$ можно построить множество деревьев вывода $Trees(A)$. На множестве деревьев вывода $\bigcup\limits_{X \in \mathcal{N} \cup \mathcal{T}} Trees(X)$ определяется функция $lts(t)$, сопоставляющая каждому дереву вывода соответствующую LTS.

В *разделе 3.4* приводится краткое описание темпоральной логики $\mathsf{CTL}^\star$ и её подмножеств CTL, LTL, $\mathsf{ACTL}^\star$-X, LTL-X, используемых для описания свойств проверяемых моделей программ. В диссертационной работе для описания свойств моделей параметризованного семейства используются преимущественно логики $\mathsf{ACTL}^\star$-X и LTL-X, в формулах которых отсутствуют оператор X и квантор существования пути.

В *разделе 3.5* вводится специальная классификация путей в моделях, которая используется в дальнейшем для определения новых разновидностей отношения симуляции между моделями. Если задана спецификация программы, то индивидуальные особенности определённых переходов LTS могут быть важны для выполнимости спецификации, в то время как особенности других переходов не важны. Переходы последнего вида считаются ненаблюдаемыми, и в моделях распределённых систем для их обозначения часто используется специальный символ τ. В общем случае для любой LTS с отношением переходов R можно выбрать некоторое множество *наблюдаемых переходов* E, $E \subseteq R$. *Конечным блоком* называется такой конечный путь $s_1 \xrightarrow{\tau} s_2 \xrightarrow{\tau} \cdots \xrightarrow{\tau} s_n \xrightarrow{a} s_{n+1}$, в котором $(s_i, \tau, s_{i+1}) \notin E$ для всех $1 \leq i < n$ и $(s_n, a, s_{n+1}) \in E$. *Бесконечным блоком* называется такой бесконечный путь $s_1 \xrightarrow{\tau} \cdots \xrightarrow{\tau} s_k \xrightarrow{\tau} \cdots$, в котором $(s_i, \tau, s_{i+1}) \notin E$ для всех $i \geq 1$. В том случае, когда в модели выделено некоторое множество наблюдаемых переменных Σ_0, то в качестве наблюдаемых переходов обычно выбираются все переходы из множества $Observ(M, \Sigma_0) = \{(s, a, t) \mid (s, a, t) \in R \text{ и } (a \neq \tau \vee L(s) \cap \Sigma_0 \neq L(t) \cap \Sigma_0)\}$, а для обозначения всех остальных переходов используется запись $s \xrightarrow{\theta} t$.

В четвёртой главе вводятся новые отношения симуляции: квазиблочная, блочная и полублочная. Исследованы основные свойства этих отношений. Показано, что введённые отношения обладают необходимым набором свойств для решения задачи верификации параметризованных моделей распределённых систем при помощи метода поиска инвариантов.

Для успешного применения этого метода инвариантов требуется выбрать отношение, удовлетворяющее свойствам монотонности для заданной операции параллельной композиции и консервативности для заданной темпоральной логики. При этом, чем слабее выбранное отношение, тем большее множество LTS находится в заданном отношении и тем проще осуществляется

поиск инварианта параметризованного семейства. В данной работе используются ослабленные отношения на моделях для поиска сетевых инвариантов среди LTS, выводимых из нетерминалов сетевой грамматики, без дополнительной модификации LTS процессов.

Среди различных отношений эквивалентности на моделях важную роль играют отношения бисимуляции. Для решения задачи PMC ранее использовалось отношение строгой бисимуляции (strong bisimulation), а также некоторые ослабленные варианты этого отношения: ветвящаяся бисимуляция (branching bisimulation), бисимуляция по прореживанию (stuttering bisimulation). Одним из ослабленных отношений эквивалентности является отношение блочной бисимуляции (block bisimulation[4]). Это отношение строже отношений ветвящейся и прореженной бисимуляции, но при этом, в отличие от ветвящейся бисимуляции, отношение блочной бисимуляции учитывает дивергенцию, т.е. бесконечные последовательности ненаблюдаемых переходов. Поэтому блочная бисимуляция обладает хорошими перспективами использования ее в качестве отношения подобия в методе поиска инвариантов.

В *разделе 4.1* вводится отношение блочной симуляции, отличающееся от отношения блочной бисимуляции отсутствием свойства симметричности.

Определение 4.1 (Блочная симуляция, bks)**.** Пусть заданы LTS $M_i = \langle S_i, S_i^0, A_i, R_i, \Sigma_i, L_i \rangle$, $i = 1, 2$, с выделенным множеством наблюдаемых переменных $\Sigma_0 \subseteq \Sigma_1 \cap \Sigma_2$. Отношение $H \subseteq S_1 \times S_2$ называется *блочной симуляцией* на моделях M_1 и M_2 относительно Σ_0 тогда и только тогда, когда для любой пары состояний $(s_1, t_1) \in H$ выполняются следующие условия:

1. $L_1(s_1) \cap \Sigma_0 = L_2(t_1) \cap \Sigma_0$.

[4]Emerson E., Namjoshi K. Reasoning about rings. *In Proceedings of 22th ACM Conf. on Principles of Programming Languages*, Pp. 85–94, 1995.

2. Для любого конечного блока $\sigma = s_1 \xrightarrow{\theta} s_2 \xrightarrow{\theta} \cdots \xrightarrow{\theta} s_m \xrightarrow{a} s_{m+1}$ из состояния s_1 найдётся такой конечный блок $\delta = t_1 \xrightarrow{\theta} t_2 \xrightarrow{\theta} \cdots \xrightarrow{\theta} t_n \xrightarrow{a} t_{n+1}$ из состояния t_1, что $(s_{m+1}, t_{n+1}) \in H$ и $(s_i, t_j) \in H$ верно для всех i, j, $1 \leq i \leq m$, $1 \leq j \leq n$.

3. Для любого бесконечного блока $\sigma = s_1 \xrightarrow{\theta} \cdots \xrightarrow{\theta} s_m \xrightarrow{\theta} \cdots$ из состояния s_1 найдётся такой бесконечный блок $\delta = t_1 \xrightarrow{\theta} \cdots \xrightarrow{\theta} t_n \xrightarrow{\theta} \cdots$ из состояния t_1, что $(s_i, t_j) \in H$ верно для всех i, j, $1 \leq i$, $1 \leq j$.

Для отношений эквивалентности вида sim и LTS M_1 и M_2 используется запись $M_1 \preceq^{\mathsf{sim}} M_2$, если найдётся такое отношение H вида sim, что для каждого начального состояния $s_0 \in S_1^0$ найдётся такое начальное состояние $t_0 \in S_2^0$, что $(s_0, t_0) \in H$. Если зафиксировано множество переменных Σ, то используется запись $M_1 \preceq_\Sigma^{\mathsf{sim}} M_2$.

Отношение блочной симуляции обладает свойством консервативности относительно формул логики $\mathsf{ACTL}^\star\text{-}\mathsf{X}$, необходимым для применения метода инвариантов:

Теорема 4.3. *Пусть даны LTS* $M_i = \langle S_i, S_i^0, A_i, R_i, \Sigma_i, L_i \rangle$, $i = 1, 2$, *находящиеся в отношении блочной симуляции* $H \subseteq S_1 \times S_2$ *относительно множества наблюдаемых переменных* $\Sigma_0 \subseteq \Sigma_1 \cap \Sigma_2$. *Для любой пары состояний* $(s, t) \in H$ *и любой формулы* φ *логики* $\mathsf{ACTL}^\star\text{-}\mathsf{X}$ *с переменными из множества* Σ_0 *верно:* $M_2, t \models \varphi \Longrightarrow M_1, s \models \varphi$.

Однако отношение блочной симуляции в общем случае не монотонно относительно операции параллельной композиции. Поэтому *в разделе 4.2* вводится более слабое отношение квазиблочной симуляции.

Определение 4.3 (Квазиблочная симуляция, qbs)**.** Пусть заданы LTS $M_i = \langle S_i, S_i^0, A_i, R_i, \Sigma_i, L_i \rangle$, $i = 1, 2$, с выделенным множеством наблюдаемых переменных $\Sigma_0 \subseteq \Sigma_1 \cap \Sigma_2$. Пусть также заданы множества наблюдаемых переходов

E_1 и E_2 моделей M_1 и M_2 соответственно. Отношение $H \subseteq S_1 \times S_2$ называется *квазиблочной симуляцией* на моделях M_1 и M_2 относительно множеств Σ_0, E_1, E_2, если для любой пары состояний $(s_1, t_1) \in H$ выполняются следующие условия:

1. $L_1(s_1) \cap \Sigma_0 = L_2(t_1) \cap \Sigma_0$.
2. Для любого конечного блока $\sigma = s_1 \xrightarrow{\theta} s_2 \xrightarrow{\theta} \cdots \xrightarrow{\theta} s_m \xrightarrow{a} s_{m+1}$ из состояния s_1 модели M_1 найдётся такой соответствующий блок $\delta = t_1 \xrightarrow{\theta} t_2 \xrightarrow{\theta} \cdots \xrightarrow{\theta} t_n \xrightarrow{a} t_{n+1}$ из состояния t_1 модели M_2, что $(s_{m+1}, t_{n+1}) \in H$ и $(s_i, t_j) \in H$ верно для всех i, j, $1 \leq i \leq m$, $1 \leq j \leq n$.
3. Для любого бесконечного блока $\sigma = s_1 \xrightarrow{\theta} \cdots \xrightarrow{\theta} s_m \xrightarrow{\theta} \cdots$ из состояния s_1 модели M_1 найдётся такой соответствующий бесконечный блок $\delta = t_1 \xrightarrow{\theta} \cdots \xrightarrow{\theta} t_n \xrightarrow{\theta} \cdots$ из состояния t_1 модели M_2, что $(s_i, t_j) \in H$ верно для всех i, j, $1 \leq i$, $1 \leq j$.

В *разделе 4.2.1* приводятся основные свойства квазиблочной симуляции. Из утверждения 4.4 следует, что отношение блочной симуляции является частным случаем отношения квазиблочной симуляции.

Утверждение 4.4. *Пусть заданы LTS* $M_i = \langle S_i, S_i^0, A_i, R_i, \Sigma_i, L_i \rangle$, $i = 1, 2$, *с множеством наблюдаемых переменных* $\Sigma_0 \subseteq \Sigma_1 \cap \Sigma_2$. *Если* $H \subseteq S_1 \times S_2$ *— отношение блочной симуляции для множества переменных* Σ_0, *то* H *— отношение квазиблочной симуляции для множеств наблюдаемых переходов* $E_1 = Observ(M_1, \Sigma_0)$ *и* $E_2 = Observ(M_2, \Sigma_0)$ *и множества наблюдаемых переменных* Σ_0.

Одним из важнейших свойств квазиблочной симуляции, необходимым для применения метода инвариантов, является монотонность этого отношения относительно операции параллельной композиции:

Теорема 4.6. *Пусть заданы:*

- *такие LTS* $M_i = \langle S_i, S_i^0, A_i, R_i, \Sigma_i, L_i \rangle$, $i = 1,2,3,4$, *что* $(\Sigma_1 \cup \Sigma_2) \cap (\Sigma_3 \cup \Sigma_4) = \emptyset$,$A_1 = A_2 = A'$, $A_3 = A_4 = A''$ *и* $A' \cap A'' = \emptyset$;
- *такие непересекающиеся множества переменных* Σ' *и* Σ'', *что* $\Sigma' \subseteq (\Sigma_1 \cup \Sigma_3)$ *и* $\Sigma'' \subseteq (\Sigma_2 \cup \Sigma_4)$;
- *синхронизатор* $\Gamma = (\Delta, ^-)$, *причём* $\Delta \subseteq A'$ *и* $^- : \Delta \to A''$.

Тогда из $M_1 \preceq^{\mathsf{qbs}}_{\Sigma'} M_2$ *и* $M_3 \preceq^{\mathsf{qbs}}_{\Sigma''} M_4$ *следует* $M_1 \parallel_\Gamma M_3 \preceq^{\mathsf{qbs}}_{\Sigma' \cup \Sigma''} M_2 \parallel_\Gamma M_4$.

Как показано в разделе 4.2.2 отношение квазиблочной симуляции является также отношением прореженной симуляции, но не наоборот.

Теорема 4.7. *Если* $M_1 \preceq^{\mathsf{qbs}}_{\Sigma_0} M_2$, *то* $M_1 \preceq^{\mathsf{sts}}_{\Sigma_0} M_2$.

Поскольку прореженная симуляция обладает свойством консервативности для формул логики $\mathsf{ACTL}^\star\text{-X}$, отсюда следует

Теорема 4.8. *Пусть для LTS* M_1 *и* M_2 *существует отношение квазиблочной симуляции, согласованное относительно формулы* φ *логики* $\mathsf{ACTL}^\star\text{-X}$, *т.е.* $M_1 \preceq^{\mathsf{qbs}} M_2$. *Если* $M_2 \models \varphi$, *то* $M_1 \models \varphi$.

Из указанных свойств следует, что отношение квазиблочной симуляции обладает всеми необходимыми свойствами для применения метода сетевых инвариантов к моделям асинхронных распределённых систем.

Главный недостаток квазиблочной симуляции — трудность вычисления этого отношения: требуется проводить тройной переборный поиск (по наблюдаемым действиям, по путям из соответствующих состояний и по всем парам состояний соответствующих путей). С точки зрения сложности поиска такого отношения выгоднее использовать отношений блочной симуляции. Для эффективного построения отношения блочной симуляции в *разделе 4.3* определяется отношение полублочной симуляции.

Определение 4.5 (Полублочная симуляция, sbs)**.** Отношение $H \subseteq S_1 \times S_2$ называется отношением *полублочной симуляции* на моделях $M_i = \langle S_i, S_i^0, A_i, R_i, \Sigma_i, L_i \rangle$, $i = 1, 2$, относительно множества наблюдаемых переменных $\Sigma_0 \subseteq \Sigma_1 \cap \Sigma_2$, тогда и только тогда, когда для любой пары состояний $(s_1, t_1) \in H$ выполняются следующие условия:

1. $L_1(s_1) \cap \Sigma_0 = L_2(t_1) \cap \Sigma_0$.
2. Для каждого конечного блока $\sigma = s_1 \xrightarrow{\theta} s_2 \xrightarrow{\theta} \cdots \xrightarrow{\theta} s_m \xrightarrow{a} s_{m+1}$ из состояния s_1 найдётся такой конечный блок $\delta = t_1 \xrightarrow{\theta} t_2 \xrightarrow{\theta} \cdots \xrightarrow{\theta} t_n \xrightarrow{a} t_{n+1}$ из состояния s_2, что
   а. если $n > 1$, то $(s_1, t_n) \in H$ и $(s_{m+1}, t_{n+1}) \in H$;
   б. если $n = 1$, то $(s_{m+1}, t_{n+1}) \in H$;
3. (дивергенция) Для каждого бесконечного блока $\sigma = s_1 \xrightarrow{\theta} \cdots \xrightarrow{\theta} s_i \xrightarrow{\theta} \cdots$ из состояния s_1 найдётся такой бесконечный блок $\delta = t_1 \xrightarrow{\theta} \cdots \xrightarrow{\theta} t_j \xrightarrow{\theta} \cdots$ из состояния t_1 и число $k > 1$, что $(s_1, t_k) \in H$.

Теорема 4.9. *Пусть даны LTS* $M_i = \langle S_i, S_i^0, A_i, R_i, \Sigma_i, L_i \rangle$, $i = 1, 2$, *и множество наблюдаемых переменных* $\Sigma_0 \subseteq \Sigma_1 \cap \Sigma_2$. *В этом случае* $M_1 \preceq^{\mathsf{sbs}}_{\Sigma_0} M_2$ *тогда и только тогда, когда* $M_1 \preceq^{\mathsf{bks}}_{\Sigma_0} M_2$.

Определение полублочной симуляции опирается на такие особенности устройства моделей, которые позволяют существенно ускорить переборный поиск — вместо тройного перебора, необходимого для вычисления квазиблочной симуляции, достаточно проводить лишь переборный поиск соответствующих блоков, исходящих из соответствующих состояний. Для осуществления этой возможности в *разделе 4.3.1* установлен эффективно проверяемый критерий существования полублочной симуляции для LTS, свободных от внутренних тупиков (livelocks).

В пятой главе описывается новая модификация метода поиска инвариантов, опирающаяся на отношения блочной, квазиблочной и полублочной симуляции, введённые в четвёртой главе. Предложен алгоритм построения отношения полублочной симуляции, с помощью которого проводится поиск инвариантов параметризованных моделей распределённых систем. Обсуждаются различные способы оптимизации практической реализации разработанного алгоритма.

Идея предложенного метода такова. Деревья вывода сетевых грамматик, при помощи которых описывается параметризованная модель распределенной системы, позволяют порождать последовательно все LTS M_k, представляющие отдельные конечные модели этой системы. Поиск подходящей модели-инварианта проводится среди LTS M_k, порождаемых деревьями вывода. Для проверки характеристических свойств инварианта в качестве отношения подобия используется отношение квазиблочной симуляции, удовлетворяющее требованиям монотонности и консервативности. Ранее доказанные теоремы гарантируют, что существование полублочной симуляции между моделями является достаточным условием существования квазиблочной симуляции между этими же моделями. Поэтому в целях повышения эффективности проверки соотношений инвариантности модели в нем вместо квазиблочной симуляции используется отношение полублочной симуляции. Если указанное соотношение выполнено и подходящая модель-инвариант M_k обнаружена, то решение задачи PMC для параметризованного семейства $\{M_i\}_{i=1}^{\infty}$ сводится к решению задачи MC для конечных моделей $M_1, M_2, \ldots, M_k$. Последняя задача решается при помощи известных алгоритмов верификации конечных моделей программ.

Более формально соотношения инвариантности модели относительно параметризованного семейства конечных моделей определяются следующим образом. Пусть задана сетевая грамматика $G = (\mathcal{T}, \mathcal{N}, \mathcal{P}, \mathcal{S})$. Пусть для каж-

дого нетерминала $A \in \mathcal{N}$ выбрано дерево вывода (представитель) $A_{repr} \in Trees(A)$. Определим функцию $repr : N \cup T \to \bigcup\limits_{X \in \mathcal{N} \cup \mathcal{T}} Trees(X)$, которая каждому нетерминалу A ставит в соответствие дерево вывода A_{repr}, а каждому терминалу $p \in \mathcal{T}$ — дерево, состоящее из самого терминала p.

Размеченные системы переходов из множества $\{lts(repr(A)) \mid A \in \mathcal{N}\}$ называются *инвариантами* соответствующих нетерминалов, если для каждого правила вывода грамматики $A \to B_1[\mathcal{R}_1] \parallel_{\Gamma_1} \cdots \parallel_{\Gamma_{n-1}} B_n[\mathcal{R}_n]$ из предпосылок

- высота дерева вывода $repr(A)$ равна h;
- и $t \in Trees(A)$ — произвольное дерево вывода из A высоты $h+1$, в котором из каждого нетерминала $B_i \neq A$ выводится дерево $repr(B_i)$;

следует соотношение $lts(t) \preceq^{\mathsf{bks}} lts(repr(A))$.

Справедлива теорема о корректности метода инвариантов

Теорема 4.10 (Корректность метода инвариантов)**.** *Предположим, что для всякого нетерминала грамматики найден инвариант $repr(A)$. Пусть $A \in \mathcal{N}$ — нетерминал грамматики, а t — произвольное дерево из множества $Trees(A)$, в котором наибольшая высота деревьев для каждого нетерминала B грамматики больше или равна высоте инварианта $repr(B)$.*

В этом случае верно соотношение $lts(t) \preceq^{\mathsf{qbs}} lts(repr(A))$.

Как следует из этой теоремы и из свойств квазиблочной симуляции, в случае существовани инвариантов для всех инвариантов сетевой грамматики для проверки выполнимости темпоральной формулы на всех моделях, порождаемых этой грамматикой, достаточно проверить выполнимость формулы на всех моделях, деревья вывода которых имеют некоторую ограниченную высоту. Поиск инвариантов проводится согласно следующей стратегии. Сначала осуществляется поиск инвариантов для тех нетерминалов A, из которых

выводятся лишь терминалы или сам нетерминал A. После того, как инварианты для таких нетерминалов найдены, осуществляется поиск инвариантов для нетерминалов, использующих в правилах вывода другие нетерминалы.

Так как поиск осуществляется только среди LTS, выводимых из нетерминалов, перебор осуществляется полностью автоматически. Однако, не для каждого параметризованного семейства процедура перебора завершается.

В *разделе 5.2* приводится краткое описание алгоритма построения полублочной симуляции с помощью итеративного вычисления неподвижной точки. Строящееся множество H делится на два непересекающихся подмножества: опровергнутые пары (negatives) и неподтверждённые пары (positives). В начале работы алгоритма пары начальных состояний первой и второй модели заносятся в множество неподтверждённых пар. На каждой итерации алгоритма проверяется гипотеза о том, что все неподтверждённые пары удовлетворяют определению полублочной симуляции. Если в ходе проверки обнаруживается, что некоторая пара не удовлетворяет этому определению, она заносится в множество опровергнутых пар. Проверка прекращается, как только множества неподтверждённых и опровергнутых пар перестаёт расширяться.

В *разделе 5.3* установлены верхние оценки сложности алгоритма. Получена следующая оценка сложности по времени алгоритма построения отношения полублочной симуляции для LTS $M_i = \langle S_i, S_i^0, A_i, R_i, \Sigma_i, L_i \rangle$, $i = 1, 2$:

$$C_{build} \leq n_1^2 \cdot n_2^2 \cdot n_A \cdot O(n_1^3 + n_2^3 + n_A \cdot n_1^2 \cdot n_2^2) \leq O(n_1^5 \cdot n_2^3 \cdot n_A^2),$$

где $n_1 = |S_1|$ — число состояний LTS M_1, $n_2 = |S_2|$ — число состояний LTS M_2, n_A — число наблюдаемых действий каждой LTS. Предполагается, что $n_A \leq n_2 \leq n_1$.

В конце раздела проводится анализ полученных оценок, а также обсуждаются некоторые особенности практической реализации и оптимизации алгоритма, связанные с выбором подходящих структур данных для представления моделей.

В шестой главе приводится описание архитектуры экспериментальной системы верификации параметризованных моделей распределённых систем. Практическая применимость разработанных методов, алгоритмов и программ верификации параметризованных моделей распределённых систем проверена на примере нескольких моделей распределённых программ и протоколов. В качестве одного из примеров была использована параметризованная модель сетевого протокола резервирования ресурсов. Приводятся результаты численного эксперимента по верификации параметризованной модели при помощи разработанной экспериментальной системы верификации.

В *разделе 6.1* описана архитектура экспериментальной системы верификации параметризованных моделей распределённых систем CHEAPS (Checker of Asynchronous Parameterized Systems). На вход системе подаются описания прототипов процессов и сетевой грамматики. По сетевой грамматике строятся модели, порождаемые нетерминальными символами грамматики. Для каждого нетерминала грамматики проводится поиск модели-инварианта среди моделей, выводимых из данного нетерминала. Проверка того, что модель является инвариантом, осуществляется с помощью построения полублочной симуляции. Когда для каждого нетерминала найдена модель-инвариант, может быть построен инвариант проверяемого семейства моделей. Далее с помощью инструментального средства верификации конечных моделей SPIN проводится проверка выполнимости спецификаций, заданных в виде формул логики LTL-X, на построенной модели-инварианте.

В тех случаях, когда найти инвариант не представляется возможным, предлагается использовать подсистему нахождения контрпримеров по построенному отношению полублочной симуляции между моделями. С помощью контрпримеров могут быть обнаружены различия в поведении моделей семейства с различным числом процессов.

В *разделе 6.2* описывается протокол резервирования ресурсов (RSVP),

который предоставляет механизм резервирования сетевых ресурсов для соблюдения заданного качества сервиса (QoS). Протокол реализован на транспортном уровне стека протоколов TCP/IP. Данный протокол используется для обеспечения определённой скорости передачи аудио- и видеотрафика от сервера к потребителю.

В *разделе 6.3* проведен обозр и краткий анализ работ, посвящённых верификации протокола RSVP. В *разделе 6.4* указаны основные абстракции, используемые при построении параметризованной модели протокола: процессы используют групповую рассылку сообщений, логическая топология системы соответствует бинарному дереву, механизмы контроля соединения и контроля доступа всегда разрешают соединение, сбои отсутствуют.

В *разделе 6.5* описаны инварианты нетерминалов сетевой грамматики параметризованной модели RSVP и выделены те LTS, для которых проводилось построение полублочной симуляции с целью вычисления этих инвариантов. Приведены численные показатели экспериментов по построению полублочной симуляции для указанных LTS, а также данные времени построения отношений симуляции и потребления оперативной памяти для различных оптимизированных вариантов алгоритма построения полублочной симуляции.

В экспериментах построение полублочной симуляции для большинства моделей требовало не более 10 сек. процессорного времени и 20 МБ памяти. Для самых больших моделей с 1,8 млн. состояний и 25 тыс. состояний блочная симуляция была построена за 11,7 ч с использованием 300 МБ памяти. В экспериментах использовался сервер Лаборатории вычислительных комплексов (процессор AMD Opteron с тактовой частотой 2.4 ГГц).

Успешно подтверждена выполнимость следующих свойств на построенной модели-инварианте протокола RSVP: производитель не получает подтверждение о закрытии сессии, если он сам не отправил сообщение о закрытии сессии; маршрутизатор не отправляет запрос на резервирование, если

для сессии уже зарезервированы ресурсы; маршрутизатор получает запрос на резервирование только после того, как был установлен маршрут; через маршрутизатор посылаются данные только в том случае, если соединение установлено.

С помощью экспериментальной системы также успешно проверялись свойства параметризованных моделей следующих распределённых алгоритмов: кольцевого планировщика Милнера, древовидного волнового алгоритма, кольцевого алгоритма двухфазной блокировки.

В **заключении** формулируются основные результаты работы и возможные направления дальнейших исследований задачи PMC и развития системы верификации СнеAPS.

В **приложении А** описываются известные отношения частичного порядка (симуляции) и эквивалентности (бисимуляции) на размеченных системах переходов, играющие важную роль в методе инвариантов, используемом для решения задачи PMC.

В **приложении Б** приводится подробное описание алгоритма построения полублочной симуляции, доказательство его корректности, обоснование оценок сложности; приводится описание экспериментальной системы верификации параметризованных моделей распределённых систем. Описываются способы оптимизации реализации алгоритмов в экспериментальной системе.

ПУБЛИКАЦИИ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Коннов И.В., Захаров В.А. Об одном подходе к верификации параметризованных симметричных распределённых программ // Труды Первой Всероссийской Научной Конференции «Методы и Средства Обработки Информации». —М.: Издательский отдел факультета ВМиК МГУ, 2003. — С. 395–400.

2. Захаров В.А., Коннов И.В. Об одном подходе к верификации асинхронных параметризованных распределённых программ // Труды Второй Всероссийской Научной Конференции «Методы и Средства Обработки Информации». —М.: Издательский отдел факультета ВМиК МГУ, 2005. — С. 367–372.

3. И.В. Коннов, В.А. Захаров. Об одном подходе к верификации симметричных параметризованных распределенных систем // Программирование. — 2005. — №5. — С. 3–17.

4. Vladimir Zakharov and Igor Konnov. An Invariant-based Approach to the Verification of Asynchronous Parameterized Networks. In International Workshop on Invariant Generation (WING'07), RISC, Hagenberg, Austria, June 25-27. RISC-Linz Report Series No. 07–07. Pp. 41–63.

5. I. V. Konnov, V. A. Zakharov. On the Verification of Asynchronous Parameterized Networks of Communicating Processes by Model Checking. // Сборник «Математические методы и алгоритмы», ИСПРАН. — 2007. — т. 12. — С. 37–58.

6. Коннов И.В. Система верификации параметризованных моделей асинхронных распределённых систем (CHEAPS) // Труды пятой Всероссийской научно-технической конференции «Технологии Microsoft в теории и практике программирования» для студентов, аспирантов и молодых ученых Российской Федерации (Центральный федеральный округ). — 2008. — С. 225-226.

7. Коннов И.В. Применение ослабленных отношений симуляции в методе сетевых инвариантов для верификации параметризованных асинхронных моделей // Моделирование и анализ информационных систем. — 2008. — т. 15, № 3. — С. 3–13.