УДК 004.432.4

СИСТЕМА АВТОМАТИЗИРОВАННОГО ДОКАЗАТЕЛЬСТВА КОРРЕКТНОСТИ ФУНКЦИОНАЛЬНО-ПОТОКОВЫХ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ

Кропачева М.С.
Научный руководитель — д.т.н., профессор Легалов А.И.
Федеральное государственное автономное образовательное учреждение высшего профессионального образования
Сибирский федеральный университет

В настоящее время идёт тенденция развития высокопроизводительных вычислений за счёт распараллеливания работы программ. Однако это приводит к появлению новых ошибок в написанных в традиционном императивном стиле параллельных программах. Напротив, разработка программ на функционально-потоковых языках параллельного программирования позволяет не только упростить разработку, но и использовать формальные методы для верификации программ.

Под формальной верификацией понимается доказательство корректности программы, которое заключается в установлении соответствия между программой и её спецификацией, описывающей цель разработки [1]. Главным преимуществом формальной верификации является возможность доказать отсутствие ошибок в программе.

Дедуктивный анализ является наиболее универсальным методом формальной верификации. В основе дедуктивного анализа лежит аксиоматический подход, основанный на исчислении Хоара [2]. Исчисление Хоара – это расширение какой-либо формальной теории, введением в неё формул специального вида, которые называются тройками Хоара. Тройка Хоара - это программа, к которой приписаны две формулы формальной теории, описывающие ограничения на входные переменные (предусловие) и требования к результату работы программы (постусловие). В результате, доказательство корректности программы сводится к доказательству истинности тройки Хоара для этой программы. Данный подход хорошо проработан и частично автоматизирован для последовательных императивных программ. Однако процесс доказательства сильно усложняется для параллельных императивных программ.

В предложенных А.И. Легаловым модели функционально-потоковых параллельных вычислений и языке программирования Пифагор, реализующем эту модель [3], отсутствуют ошибки, характерные для императивных параллельных программ. Поэтому процесс формальной верификации по сложности сравним с доказательством корректности последовательной программы.

Для функционально-потокового языка параллельного программирования Пифагор разработана аксиоматическая теория, позволяющая проводить доказательства корректности программ. Для каждой встроенной функции языка задаётся набор аксиом, описывающих работу этой функции, также используется два правила вывода: «правило прямого прослеживания» и «правило преобразования в формулу». «Правило прямого прослеживания» позволяет на основе аксиом для встроенных функций преобразовывать тройки Хоара любой программы на языке Пифагор, так как любая программа, в конечном счёте, состоит из встроенных функций. Применение правила приводит к «сокращению» («свёртке») программы и связанному с этим изменению пред- и постусловия. После последовательных применений «правила прямого прослеживания» получается тройка Хоара с «пустой программой», которая переводится в формулу с помощью «правила преобразования в формулу». Далее можно доказывать истинность формулы, из которой будет следовать истинность программы.

Однако процесс доказательства функционально-потоковых программ, также как императивных последовательных программ, достаточно трудоёмок, так как требует рассмотрения большого числа вариантов выполнения программы и работы с громоздкими формулами. Поэтому целесообразно автоматизировать данный процесс.

Так как проблема корректности программ алгоритмически неразрешима, нельзя полностью автоматизировать процесс доказательства корректности программы. Этапы определения тройки Хоара для конкретной программы и доказательства условий корректности не могут быть автоматическими, а должны проводиться в диалоговом режиме. Этап генерации условий корректности, напротив, может выполняться автоматически [1].

В общем виде систему, позволяющую автоматизировать доказательство корректности функционально-потоковых параллельных программ на языке Пифагор, можно представить в виде схемы, приведённой на рисунке 1.

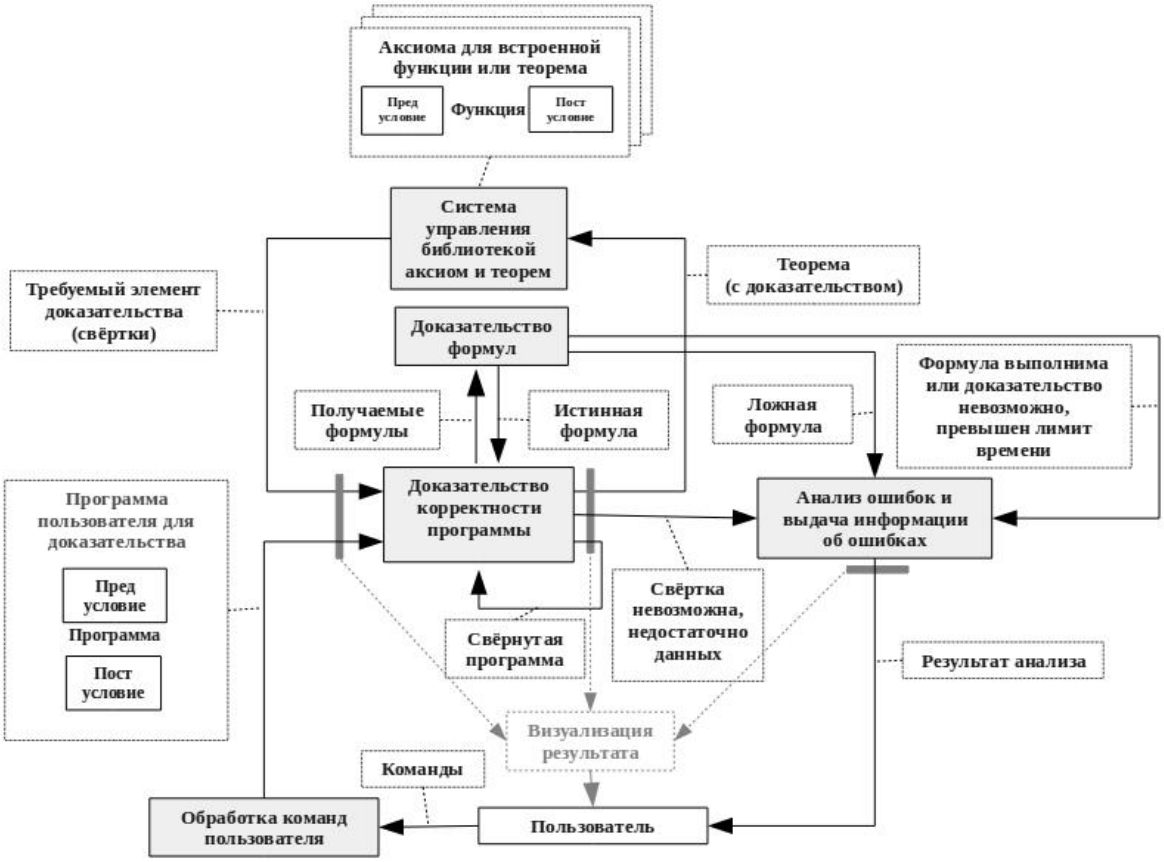

Рисунок 1. Общая схема системы автоматизированного доказательства корректности функционально-потоковых параллельных программ.

Пользователь передаёт системе программу на языке Пифагор и требования к программе в виде тройки Хоара (например, в текстовом виде). После считывания и перевода во внутренний формат, тройка визуализируется в виде информационного графа. Пользователь может изменять программу, пред- и постусловие, а также запустить процесс доказательства.

После запуска доказательства начинает работать блок «Доказательства корректности программы».

Основная задача блока - преобразовать («свернуть») программу в формулу, истинность которой будет доказывается с помощью блока «Доказательства формул». Данный блок выполняет преобразования программы последовательным применением

«правил прямого прослеживания» и «правила преобразования в формулу», после получения тройки с «пустой программой».

«Системы управления библиотекой аксиом и теорем» хранит аксиомы и теоремы. Аксиомами, как было описано выше, являются тройки Хоара для встроенных функций. Теоремы - это тройки Хоара пользовательских программ, корректность которых была доказана.

Для того чтобы применить «правило прямого прослеживания» блок «Доказательства корректности программы» должен определить, какая функция применяется непосредственно к аргументу, и получить от «Системы управления библиотекой аксиом и теорем» аксиому для этой функции, если она встроенная, или теорему, если она определена пользователем. Если аксиомы или теоремы отсутствуют, то «свёртка» невозможна. Пользователь должен либо предоставить аксиому, либо, в начале, доказать корректность требуемой функции и получить теорему. Если необходимые аксиомы и теоремы присутствуют в библиотеке аксиом и теорем, «Система управления библиотекой» передаёт их блоку «Доказательства корректности программы».

Из всех полученных аксиом и теорем блок «Доказательства корректности программы» должен выбрать только те, которые описывают реализуемые пути выполнения программы, и исключить заведомо недостижимые. При этом достижимыми будут только те пути выполнения программы, у которых предусловие аксиом может следовать из предусловия программы. Поэтому блок «Доказательства корректности программы» формирует формулы (их количество соответствует числу аксиом для рассматриваемой функции), описывающие истинность условия недостижимости, и передает их блоку «Доказательства формул». Блок «Доказательства формул» пытается проверить истинность формул и в результате своей работы для каждой формулы сообщает, что формула истинна, ложна, выполнима или выдаёт ошибку: «Доказательство невозможно, превышен лимит времени». В последнем случае пользователь должен определить «выполнимость» формулы самостоятельно. После получения информации об истинности формул блок «Доказательства корректности программы» отбрасывает все аксиомы соответствующие истинным формулам и применяет «правило прямого прослеживания» на основе оставшихся аксиом. В результате программа «сокращается», а из исходной тройки Хоара получается несколько новых троек, число которых соответствует числу оставшихся аксиом. Далее процесс применения «правила прямого прослеживания» повторяется для каждой из полученных троек до тех пор, пока не останутся только тройки с «пустой программой».

Процесс последовательных преобразований тройки Хоара, при применении «правил прямого прослеживания», отображается на экране пользователя. При необходимости, пользователь может переключаться между разными этапами преобразований и тройками Хоара каждого этапа.

Полученные в результате преобразований тройки с «пустой программой», преобразуются в формулы (по «правилу преобразования тройки в формулу») блоком «Доказательства корректности программы» и передаются блоку «Доказательства формул», который для каждой формулы должен сообщить, что формула истинна, ложна, выполнима или выдать ошибку: «Доказательство невозможно, превышен лимит времени». В последнем случае пользователь должен сам установить, истинна формула или нет.

Программа будет корректна, если все формулы истины. В этом случае блок «Доказательства формул» сообщит о том, что все формулы истинны блоку «Доказательства корректности программы», тогда доказанная теорема будет помещена в библиотеку теорем и аксиом «Системой управления библиотекой аксиом и теорем».

Если не все формулы истинны, то сообщение об этом передаётся блоку «Анализа ошибок и выдачи информации об ошибках». Этот блок анализирует полученную информацию. Если была найдена ложная или выполнимая формула, то в программе или спецификации содержится ошибка. Блок анализа ошибок может отобразить пользователю те пути выполнения программы, которым соответствуют ложные или выполнимые формулы. Пользователь может просмотреть последовательность преобразований каждого пути, внести изменения в программу и заново запустить процесс доказательства.

Рассмотренная система автоматизированного доказательства корректности функционально-потоковых параллельных программ, позволит упростить и частично автоматизировать процесс доказательство корректности функционально-потоковых параллельных программ.

Список литературы.

1. Непомнящий В.А. Прикладные методы верификации программ [Текст]: науч. изд. / В.А. Непомнящий, О.М. Рякин. - М.: Радио и связь, 1988. – 255 с.

2. Hoare C. A. R. An axiomatic basis for computer programming / C. A. R. Hoare // Communications of the ACM. - 1969. - Vol. 10. -No 12. - P. 576–585.

3. Легалов А.И. Функциональный язык для создания архитектурно-независимых параллельных программ // Вычислительные технологии. 2005. № 1 (10). С. 71-89.