UDC 004.4'23

Development Modules for Specification of Requirements for a System of Verification of Parallel Algorithms

[1] Vasiliy Yu. Meltsov
[2] Gennadiy A. Chistyakov

[1] Vyatka State University, Russia
Russia 610000, Kirov, st. Moscow, 36
PhD (Engineering), Associate professor
E-mail: meltsov69@mail.ru
[2] Vyatka State University, Russia
Russia 610000, Kirov, st. Moscow, 36
PhD student
E-mail: gennadiychistyakov@gmail.com

Abstract. This paper presents the results of the development of one of the modules of the system verification of parallel algorithms that are used to verify the inference engine. This module is designed to build the specification requirements, the feasibility of which on the algorithm is necessary to prove (test).

Keywords: verification; parallel algorithm; temporal logic of linear time; a top-down predictive parser.

В настоящее время все более актуальной становится проблема верификации алгоритмов функционирования сложных программных управляющих систем. Ошибки в работе таких систем могут привести к огромным финансовым потерям и даже к человеческим жертвам. Особые трудности вызывает верификация параллельных алгоритмов. В этом случае возникает целый класс новых ошибок, связанных с взаимодействием потоков друг с другом. Тестирование характеризуется сравнительно небольшими накладными расходами, но в силу непредсказуемости поведения параллельных алгоритмов, совершенно не подходит для верификации систем, к которым предъявляется требование высокой отказоустойчивости. Даже самое глубокое тестирование не всегда может выявить ошибки в работе таких алгоритмов, так как они могут не проявляться до наступления специфических условий.

В последнее время заметна тенденция к продвижению формальных методов верификации параллельных алгоритмов. Формальная верификация зачастую требует больших аппаратных и временных ресурсов, однако только формальные методы позволяют однозначно установить факт отсутствия тонких ошибок в алгоритмах работы сложных управляющих устройств.

В разрабатываемой программной системе в качестве базового метода формальной верификации используется наиболее перспективный, с точки зрения компьютерной реализации, метод проверки моделей – model checking [1], включающий в себя три основных этапа: построение модели алгоритма, спецификацию требований и проверку выполнимости требований на модели.

Структура предлагаемой системы верификации параллельных алгоритмов изображена на рисунке 1.

Первый модуль системы предназначен для построения модели верифицируемого алгоритма. В результате преобразований, выполняемых в данном блоке, осуществляется построение адекватной алгоритму структуры Крипке, формирование которой происходит в соответствии с описанным специальным образом алгоритмом и выделенными существенными свойствами данного алгоритма [2]. Такие свойства могут варьироваться в весьма широком диапазоне. Например, «может ли произойти проверка данного условного оператора после выполнения указанного оператора присваивания?».

Второй модуль системы позволяет определить проверяемое требование. Классической логики недостаточно для описания поведения сложных систем во времени, поэтому для

спецификации требований используется расширение логики высказываний – темпоральная логика линейного времени (LTL). Использование LTL позволяет жестким образом задавать порядок событий во времени, что крайне важно при поиске ошибок в параллельных алгоритмах.

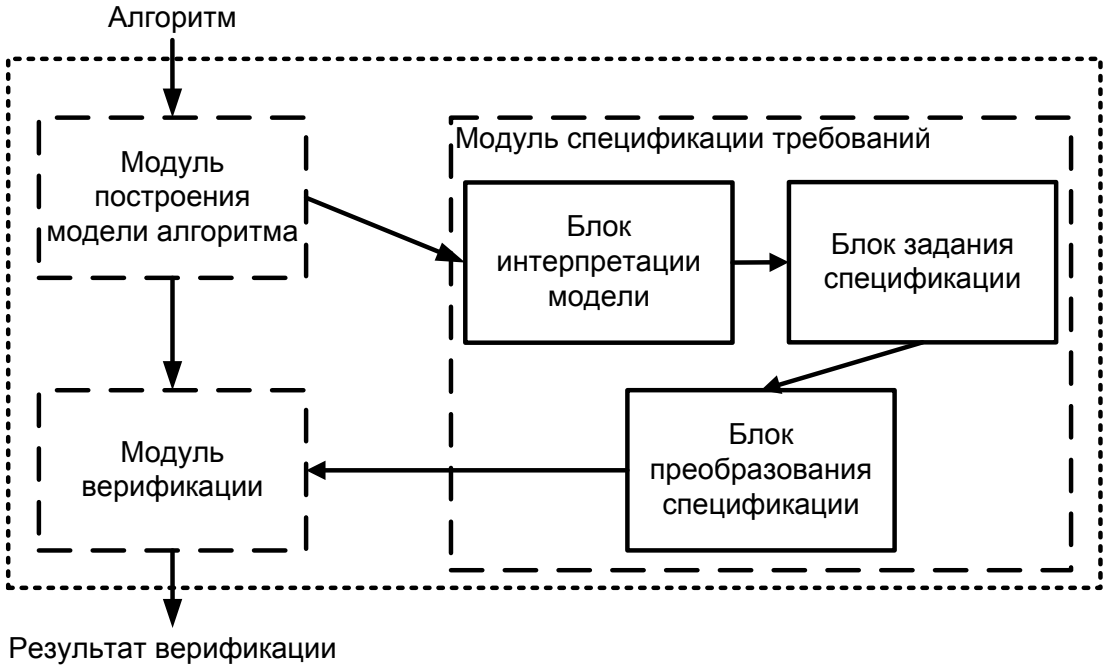

Рис. 1. Структура системы верификации

Третий модуль системы предназначен для проверки соответствия модели алгоритма специфицированному требованию. Проверка выполняется с помощью дедуктивного метода логического вывода и заключается в доказательстве гипотезы, существует ли на заданной модели такая траектория поведения, при которой заданная специфицируемая формула выполнима.

Рассмотрим подробнее принципы функционирования модуля спецификации требований, который включает в себя три основных узла.

Блок интерпретации модели предназначен для выделения из построенной модели свойств, которые могут быть специфицированы. Каждому свойству, заданному на этапе построения модели, ставится в соответствие атомарный предикат, истинное значение которого эквивалентно выполнимости свойства.

Блок задания спецификации позволяет сформировать логическое условие (спецификацию), выполнимость которого требуется проверить. Спецификация задается с помощью логических операторов (конъюнкция, дизъюнкция, отрицание), модальных операторов темпоральной логики линейного времени и атомарных предикатов, соответствующих свойствам модели. Допустимыми темпоральными операторами являются операторы «в следующий момент» (X), «всегда» (G), «в будущем» (F) и «когда-нибудь в будущем, а пока» (U) [3].

Таким образом, грамматика спецификации LTL может быть определена с помощью множества терминальных символов (все атомарные предикаты, скобки, логические и темпоральные операторы), множества нетерминальных символов (формула, слагаемое, множитель и токен), а также форм Бэкуса-Наура, представленных на рисунке 2.

$$< formula >::=< summand > \{\vee < formula >\}$$
$$< summand >::=< factor > \{\vee < formula >\}$$
$$< factor >::= \{\neg\} < token >$$
$$< token >::= \{F \mid G \mid X\}(< formula >) \mid (< formula >)U(< formula >) \mid < atomic >$$

Рис. 2. Формы Бэкуса-Наура для выражений LTL

Блок преобразования спецификации выполняет лексический анализ заданного логического условия и построение эквивалентного ему дерева грамматического разбора.

Построение дерева разбора возможно с помощью алгоритма предиктивного нисходящего анализа, состоящего из следующих шагов.

1. Проверить, что анализируемое логическое условие задано корректным образом (является правильной скобочной последовательностью, содержит только известные терминальные символы и т.д.).

2. Сопоставить дереву разбора вершину – корень. Перейти к процедуре обработки нетерминала $< formula >$.

3. Выделить следующую лексему, а если такой нет – перейти к п. 5.

4. Выполнить обработку соответствующего нетерминала. Если требуется выбрать очередную лексему, то перейти к п. 3, иначе выполнить п. 4 еще раз.

5. Алгоритм завершен.

После построения дерева разбора спецификации необходимо провести редукцию лишних вершин дерева. Так, например, вершина дерева, соответствующая операции конъюнкции и имеющая только одну вершину-потомка, может быть сокращена.

Будем называть длинной формулы количество лексем в ней. Тогда:

– выделение лексем заданного выражения может быть выполнено за линейное время с помощью тривиального алгоритма лексического анализа;

– проверка корректности формулы может быть осуществлена за время, пропорциональное длине формулы, с применением стекового конечного автомата;

– так как при построении дерева разбора каждая лексема анализируется ровно один раз, то данный этап может быть выполнен за линейное время;

– редукция вершин дерева также потребует линейного времени.

Для хранения выделенных лексем понадобится количество памяти, пропорциональное длине формулы. Стековый конечный автомат в худшем случае также потребует линейного объема памяти. Количество вершин в построенном дереве разбора не превысит длины формулы.

Таким образом, представленный алгоритм построения дерева грамматического разбора заданной спецификации характеризуется линейным временем работы и затратами на хранение данных, пропорциональными длине спецификации.

Предлагаемый модуль программного комплекса верификации параллельных алгоритмов предназначен для спецификации требований к верифицируемому алгоритму и позволяет на основе некоторых существенных свойств модели алгоритма определить логическое условие, выполнимость которого следует доказать. Построенное с помощью оригинального алгоритма дерево грамматического разбора вместе с моделью передается для последующей проверки корректности алгоритма в модуль верификации.

В отличие от привычного тестирования описываемый подход позволяет однозначно установить корректность или некорректность алгоритма. Такой результат становится возможным благодаря рассмотрению всевозможных цепочек состояний алгоритма и поиску контрпримера – одной из цепочек, приводящих к ошибочному состоянию системы (взаимоблокировка потоков, ошибка синхронизации по доступу к данным и т.д.).

Примечания:

1. Карпов Ю.Г. Model Checking. Верификация параллельных и распределенных программных систем. СПб.: БХВ-Петербург, 2010. 560 с.

2. Мельцов В.Ю., Чистяков Г.А. Разработка модуля построения структуры Крипке для системы верификации параллельных алгоритмов // Научно-технический вестник Поволжья. 2011. №6. С. 217-220.
3. Baier C. Principles of Model Checking / C. Baier, J.-P. Katoen. Cambridge: The MIT Press, 2008. 994 c.

УДК 004.4’23

Разработка модуля спецификации требований для системы верификации параллельных алгоритмов

[1] Василий Юрьевич Мельцов
[2] Геннадий Андреевич Чистяков

[1] Вятский государственный университет, Россия
610000, Киров, ул. Московская, 36, кандидат технических наук, доцент
E-mail: meltsov69@mail.ru
[2] Вятский государственный университет, Россия
610000, Киров, ул. Московская, 36, аспирант
E-mail: gennadiychistyakov@gmail.com

Аннотация. В работе представлены результаты разработки одного из модулей системы верификации параллельных алгоритмов, использующей для верификации аппарат логического вывода. Данный модуль предназначен для построения спецификации требования, выполнимость которого на исследуемом алгоритме необходимо доказать (проверить).

Ключевые слова: верификация; параллельный алгоритм; темпоральная логика линейного времени; предиктивный нисходящий анализатор.