УДК 004.052.42

МОДЕЛЬ УСЛОВНОЙ ПЕРЕМЕННОЙ ПРИ ВЕРИФИКАЦИИ ПАРАЛЛЕЛЬНОЙ ПРОГРАММЫ

О. С. Крюков[1], А. Н. Ивутин[2]
Тульский государственный университет, пр. Ленина, 92, 300012, г. Тула, Россия,
[1]ol_kryukov97@mail.ru, [2]alexey.ivutin@gmail.com

Рассматривается проблема верификации параллельных программ. Приводится описание основных примитивов синхронизации параллельной программы. Описывается модель параллельной программы с использованием условной переменной в формате расширенной сети Петри с семантическими связями.

Ключевые слова: *параллелизм, формальная верификация, проверка эквивалентности, сети Петри, модель, условная переменная.*

CONDITIONAL VARIABLE MODEL IN VERIFICATION OF A PARALLEL PROGRAM

O. S. Kryukov[1], A. N. Ivutin[2]
Tula State University, ave. Lenin, 92, 300012, Tula, Russia,
[1]ol_kryukov97@mail.ru, [2]alexey.ivutin@gmail.com

The problem of verification of parallel programs is considered. A description of the main synchronization primitives of a parallel program is given. A model of a parallel program using a conditional variable in the format of an extended Petri nets with semantic relations.

Keywords: *parallelism, formal verification, equivalence check, Petri nets, model, conditional variable.*

Современные вычислительные системы предоставляют широкие возможности по организации параллельных вычислений за счет своей многопроцессорной и/или многоядерной архитектуры. Благодаря такому подходу можно кратно увеличить производительность при выполнении программного кода по сравнению с последовательным вариантом. В связи с этим все больше современного программного обеспечения разрабатывается с учетом возможного параллелизма для наиболее эффективного использования вычислительных мощностей.

Однако обеспечение корректности и надёжности работы таких систем представляет собой сложную задачу, поскольку «плавающий» характер ошибок, связанных с применением технологий распараллеливания, делает невозможным быстрое их обнаружение. Зачастую единственным способом обнаружить такую ошибку являются многочисленные пробные запуски с целью отследить их возникновение. Подобное решение не очень удобно и может быть крайне затруднено в случае работы с крупной информационной системой или системами, требующими большого объема вводимых вручную данных, все комбинации которых проверить невозможно. В связи с этим особое значение принимают различные системы, способные выполнять верификацию параллельного программного обеспечения в автоматизированном режиме.

К таким системам относятся статические верификаторы [1], выполняющие оценку корректности программы без реального выполнения исходного кода путём проверки выполнимости различных правил на её модели. Таким образом, одним из этапов выполнения статической верификации, является построение модели программы, при котором особое внимание следует уделить следующим конструкциям, обеспечивающим взаимоисключающий доступ к общим данным:

- атомарные операции, выполняемые за один раз либо вообще не выполняемые;
- барьер, представляющий собой блокировки программы, пока все её параллельные части не достигнут заданной точки;
- мьютекс, обеспечивающий взаимное исключение критических участков кода;
- условная переменная, выполняющая блокировку одного или нескольких потоков до момента поступления сигнала от другого потока о выполнении некоторого условия или до истечения максимального промежутка времени ожидания;
- монитор, представляющий специальный тип данных, аналогичный классу или объекту в ООП, служащий для разграничения доступа к критическому коду программы.

В данной работе рассмотрим построение модели условной переменной, как наиболее комплексного инструмента, расширяющего возможности мьютекса и имеющего более сложную структуру, что усложняет её моделирование. В качестве инструмента моделирования будем применять расширенные сети Петри с семантическими связями (РСПСС) [2; 3]. В зависимости от особенностей технологии параллельного программирования возможны небольшие особенности моделирования работы с условными переменными. Рассмотрим реализацию, предлагаемую технологией std::thread языка C++.

Для использования условной переменной должен быть хотя бы один поток, ожидающий, пока какое-то условие станет истинным. Ожидающий поток должен сначала выполнить unique_lock. Эта блокировка передается методу wait(), который выполняет проверку условия и либо освобождает мьютекс и приостанавливает поток, пока не будет получен сигнал от условной переменной о повторной проверке, если условие не выполнено, либо выполняет работу, ограниченную условной переменной. Должен быть хотя бы один поток, сигнализирующий о том, что необходимо осуществить повторную проверку.

Фрагмент РСПСС для моделирования условной переменной представлен ниже (рис.).

Так как unique_lock не является атомарной операцией, то она представляется несколькими операциями (позиции 6–8 на рисунке), а для отображения состояния unique_lock вводится особая позиция (lock на рисунке), которая в заблокированном состоянии не содержит метки, наличие же метки в данной позиции означает возможность блокировки условной переменной. Операция wait() в свою очередь вызывает функцию проверки условия, поэтому в модели она заменяется операциями этой функции (позиции 9–11), а также позицией ожидания получения уведомления о возможности повторной проверки (позиция 13), которая является входом для перехода 18, ведущего в позицию начала блокировки unique_lock (позиция 7), и второй входной позицией которого является позиция notification (позиция 3), получающая метку после вызова notify_one() (позиция 3).

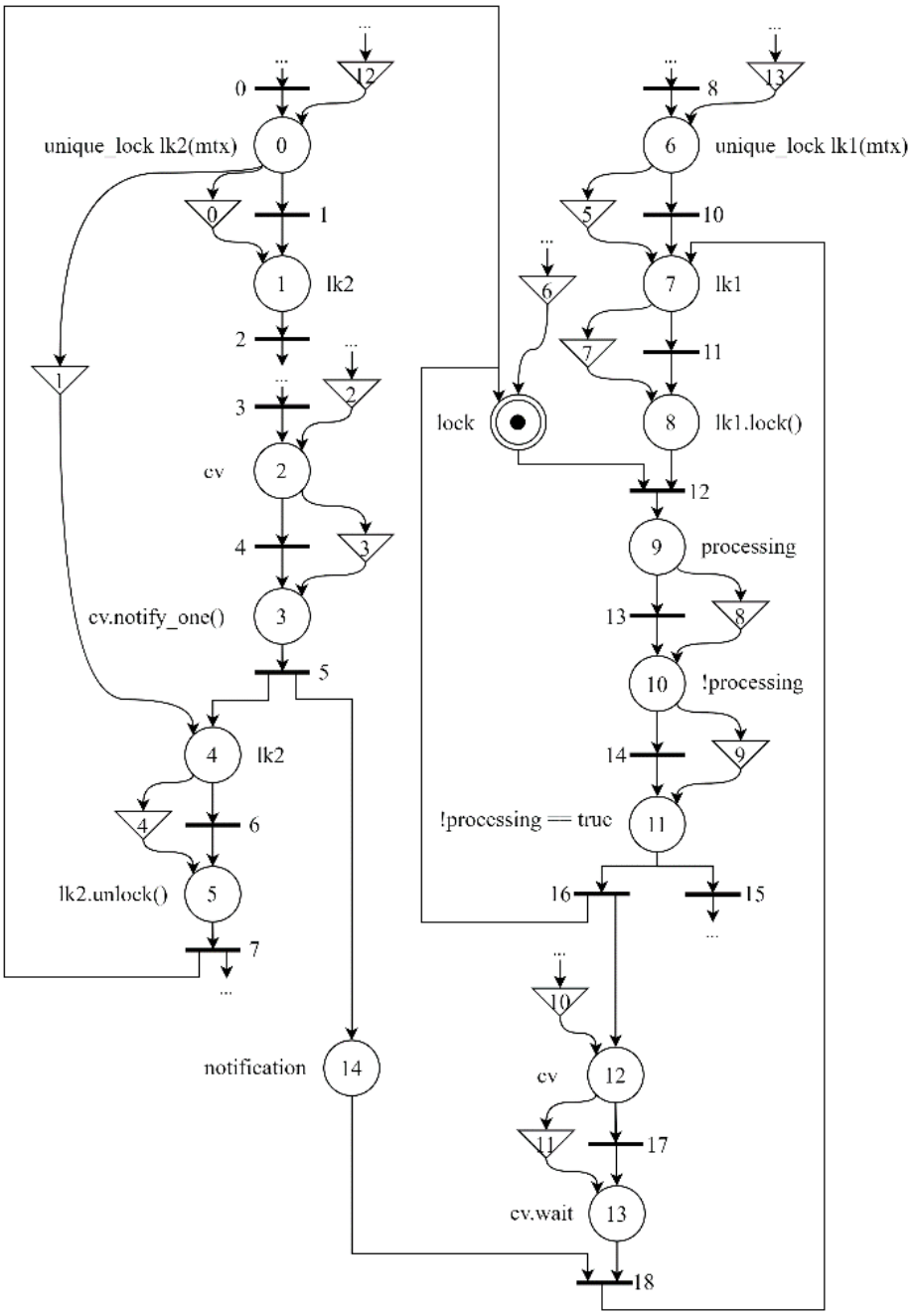

Рис. Фрагмент модели для программы с условной переменной

Применение данной модели в статическом верификаторе параллельной программы позволит проводить поиск тупиковых ситуаций, связанных с некорректным применением условной переменной, а с помощью дополнительного анализа семантических связей находить гонки.

Библиографический список

1. Мордань В. О. Комбинация методов статической верификации композиции требований // Труды ИСП РАН. 2017. № 3. 2017. С. 151–170.
2. Ivutin A. N., Voloshko A. G.. Method of Formal Verification of Program Code based on Petri Net with Additional Semantic Relations. 2020 ELEKTRO. 2020. P. 1–6.
3. Крюков О. С. Статический метод обнаружения гонок данных на основе анализа модели расширенной сети Петри // Оптико-электронные приборы и устройства в системах распознавания образов и обработки изображений. Распознавание-2023: сб. материалов XVII Междунар. науч.-техн. конф. г. Курск, 2023. С. 131–133.