УДК 004.233

ОБЪЕДИНЕННЫЙ АЛГОРИТМ ФАЗЗИНГ ТЕСТИРОВАНИЯ И ТОЧНОГО ОПРЕДЕЛЕНИЯ ВЗАИМОБЛОКИРОВОК

Доронин О.В., Мазунин К.Ю., Дергун К.И.
Университет ИТМО, г. Санкт-Петербург

Научный руководитель – к.т.н., доцент Дергачев А.М.
Университет ИТМО, г. Санкт-Петербург

В работе рассматривается подход, объединяющий возможности двух алгоритмов – алгоритма фаззинг тестирования и алгоритма явного определения взаимоблокировок. Данный подход опробован и реализован в Google Thread Sanitizer.

Введение. Одной из популярных моделей разработки программных систем является сервисная архитектура, которая позволяет запускать программные приложения на нескольких компьютерах и дает возможность как вертикального, так и горизонтального масштабирования, что повышает производительность. К сложностям разработки и тестирования таких программных систем добавляется человеческий фактор – разработка ведется параллельно большими группами разработчиков, что увеличивает вероятность появления ошибок в программном коде, например: взаимоблокировки, проблема ABA, не валидное освобождение ресурсов, гонки данных.

Основная часть. Для поиска ошибок в многопоточном коде существуют различные инструменты, такие как valgrind, google thread sanitizer, intel vtune, имеющие свои преимущества и недостатки. Так, например, в google thread sanitizer до недавнего времени отсутствовала возможность фаззинг тестирования потоков, не было возможности явного определения взаимоблокировок и был реализован только вероятностный алгоритм на графе. В данной работе предлагается подход, объединяющий возможности двух алгоритмов – фаззинг тестирования и алгоритма явного определения взаимоблокировок. Алгоритм определения взаимоблокировок отслеживает попытку захвата мьютекса, захват мьютекса, освобождение мьютекса. Алгоритму необходимо хранить список захваченных мьютексов для каждого потока captured [T, mutlist], где T – поток, a mutlist – список захваченных мьютексов в потоке T. Также для каждого потока необходимо хранить попытку захвата мьютекса try [T, M], где T – поток, а M – захваченный мьютекс. После чего необходимо искать путь на графе. Алгоритм фаззинг тестирования должен перехватывать различные точки синхронизации и осуществлять в них управление порядком выполнения потоков.

Выводы. Апробация данного подхода с использованием Google Thread Sanitizer показала, что объединенный алгоритм не имеет ложных срабатываний и позволяет не только с большей вероятностью находить взаимоблокировки, но и явно указывать на их местоположение в программном коде.

Доронин О.В. (автор) _____

Дергун К.И. (соавтор) _____

Мазунин К.Ю. (соавтор) _____

Дергачев А.М. (научный руководитель) _____