УДК: 004.233

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

VIII Конгресс молодых ученых (КМУ)

Фаззинг тестирование fined-grained алгоритмов

Авторы: *Дергун К.И., Доронин О.В., Дергачев А.М., Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики (ИТМО), г. Санкт-Петербург*

Научный руководитель: *Дергачев А.М., Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики (ИТМО), г. Санкт-Петербург*

Большинство современных программных приложений являются многопоточными, что позволяет увеличить производительность за счет параллельного выполнения задач. В многопоточных приложениях возможны следующие типы ошибок: гонки данных, взаимоблокировки, инверсии приоритетов, проблема ABA и другие. Ошибки могут приводить к крупным финансовым потерям, например, в банковской инфраструктуре, или же потерям человеческих жизней в самолетостроении, медицинских приборах, машиностроении и других областях. Для уменьшения вероятности появления ошибок существуют специальные программы, такие как valgrind и Google TSAN (thread sanitizer). Но такие инструменты не позволяют проводить фаззинг тестирование приложений.

Техника фаззинга в наши дни широко распространнена. Фаззинг техника тестирования программного обеспечения, часто автоматическая или полуавтоматическая, заключается в передаче приложению на вход неправильных, неожиданных или случайных данных. Предметом интереса являются падения и зависания, нарушения внутренней логики и проверок в коде приложения, утечки памяти, вызванные такими данными на входе. Но информацию о применении такого подхода в случае многопоточных алгоритмов, таких как fained-grained, найти сложно.

Fined-grained алгоритмы – это методы синхронизации, как правило, построенные не на применении примитивов синхронизации, предоставляемых ОС, а на применении “легких” атомарных примитивов, например, spin-lock (возможно частичное применение примитивов синхронизации ОС). На основе таких примитивов строятся структуры данных, допускающие параллельное чтение или даже параллельную запись, в которых синхронизация производится на уровне узла (node) или страницы (page, bucket) структуры данных и встроена в сами алгоритмы операций над этой структурой. Часто fine-grained контейнеры показывают производительность, сравнимую с производительностью lock-free контейнеров при относительно небольшой нагрузке. Например, библиотека libcds поддерживает не только lock-free алгоритмы, но еще и fined-grained.

В целом для фаззинг тестирования lock-free алгоритмов существует библиотека Relacy Race Detector (RRD). Это инструмент для эффективного выполнения юнит-тестов для алгоритмов синхронизации, написанных на C++0x. Каждый пользовательский поток представлен в виде fiber. При исполнении программы в каждый момент времени работает только один fiber, а специальные планировщики управляют выбором порядка исполнения fiber. Такой инструмент обладает рядом недостатков, часть которых была исправлена при переносе инфраструктуры в Google TSAN. Некоторые результаты были представлены в работе «Анализ проблем в Relacy Race Detector с последующим устранением» Доронин О.В., Калишенко Е.Л., но они относятся к lock-free алгоритмам и не приспособлены для алгоритмов fined-grained.

В данной работе получены результаты, которые позволят тестировать fined-grained алгоритмы, а также проведен сравнительный анализ алгоритмов для поддержания блокирующих примитивов синхронизации в библиотеке TSAN.

Авторы: _____ /Дергун К.И./ _____ /Доронин О.В./

Руководитель: _____ /Дергачев А.М./