Здравствуйте!

Меня зовут Леонид Сенюков, я являюсь студентом 4 курса математико-механического факультета СПбГУ и 2 курса Санкт-Петербургского отделения Computer Science Center.

Опыт программирования на C++ показал мне, насколько одновременно сложным и важным этапом разработки программного обеспечения является его тестирование. Даже в случаях однопоточных программ тестами не всегда удаётся покрыть все сценарии работы алгоритмов, не говоря уже о многопоточных приложениях.

В моей практике неоднократно возникали ситуации (особенно при первых попытках написания многопоточного кода), когда приходилось отлаживать код на C++ при помощи helgrind/drd. Разумеется, в то время я не задумывался о том, как устроены эти утилиты и как происходит процесс контроля работы приложения.

Почти сразу после начала использования утилит мною было подмечено несколько недостатков:
- Невозможность тестирования lock-free алгоритмов (утилиты не знали даже про безопасность std::atomic)
- Утилиты работали только как фон для собственноручно написанных тестов, иногда необходимо было прогонять в цикле программу на одном и том же тесте несколько раз, чтобы получить шанс поймать что-то неладное

Одним из следствий выявленных неудобств явилась, например, невозможность протестировать реализованную мною структуру данных, необходимую для исследовательского проекта.

Проект, над которым я работал, был посвящён исследованию и реализации разных подходов для решения задачи построения персистентного ландшафта вероятностным методом (когда из облака точек делаются выборки, на которых считаются определенные его характеристики, а затем результаты особым образом усредняются). В ходе работы мне было необходимо реализовать многопоточное симплициальное дерево — структуру данных, которая была бы потокобезопасной и позволяла бы хранить информацию о симплексах и производить с ними определенные операции.

В процессе работы мной было реализовано несколько версий структуры, в том числе с использованием библиотеки tbb, из которой мне была нужна tbb::unordered_map для узлов дерева. К моему удивлению, при запуске тестов под утилитой helgrind количество ошибок было чрезвычайно велико. Оказалось, что helgrind не знает о безопасности контейнера, как не знает и о безопасности std::atomic.

Имея под рукой удобный инструмент для тестирования структуры данных, я бы сэкономил уйму времени, сил и нервов. Именно поэтому я бы хотел принять участие в данном проекте, который в будущем смог бы помочь сотням тысяч программистов, которые могут столкнутся с похожей проблемой.

Разумеется, кроме «научного» интереса, мною движет и более практический — мне кажется, участие в таком проекте будет очень выгодно смотреться в моем резюме!