Concurrent Data-Structures Through Explicit Timestamping

> Trello
> libcds-fork

Перечень доработок

1. Удаление неиспользуемой памяти
2. Избавиться от ограничения количества потоков
3. Использование различных моделей памяти(?)

Вопросы

1. Стоит ли реализовавывать контейнер на RCU вместо HP?
    1. *[khizmax] Хорошо бы иметь обе - для RCU и для HP. Это две совершенно разные техники, требуют двух разных специализаций. Для RCU писать проще, но там могут возникнуть сложности с физическим удалением - внутри критической секции RCU вызывать retire_ptr (удаление элемента) нельзя, будет deadlock*
2. Избавление от зависимости количества HP заданных в системе. Сравнение производительности с HP реализацией.
    1. *[khizmax] Вот тут я не понял. Избавление от кол-ва HP - это переписать HP. Задача интересная, хватит ли времени?*
3. Стоит ли включать в текущую систему тестирования libcds, или использовать Google Tests/аналоги.
    1. *[khizmax] Использовать Google Test. В dev-ветке unit-тесты уже на 80% переведены на gtest (попутно найдено немало ошибок), создана инфраструктура для stress (многопоточных) тестов, stress-тесты для очереди и стека переведены на gtest*
4. Как правильно производить тестирование дека? Какую статистику было бы полезно собирать? Проваленные операции извлечения, количество извлечений, количество добавлений?
    1. *[khizmax] Тестировать как очередь (2 реализации - лево- и правосторонняя) и как стек (тоже 2 реализации).*
    2. *[khizmax] Кол-во извлечений/добавлений слева/справа + кол-во интересных случаев, например, сколько раз не удалось добавить с первого раза (contention на CAS). Вообще это видно по алгоритму, когда его напишешь и начинаешь отлаживать*
5. Имеет ли смысл попробовать использовать тестирование алгоритма через CB-DPOR?
    1. *[khizmax] Безусловно имеет. Тут я не могу что-либо порекомендовать, так как не использовал его. Но если получиться как-то это прикрутить - здорово. Также обратить внимание на threadSanitizer - он уже есть из коробки (gcc, clang), использовать его просто (компиляция со спец. ключами), потом ломать голову над тем, что он нашел*

From:
http://wiki.osll.ru/ - **Open Source & Linux Lab**

Permanent link:
http://wiki.osll.ru/doku.php/projects:libcds:timestamp_structures

Last update: **2016/05/22 10:44**