2025/11/05 11:17 1/2 Lock-free контейнер

Lock-free контейнер

Необходимо реализовать в lock-free стиле следующий интерфейс:

```
/**
* Lock-Free множество.
* @param <T> Тип ключей
*/
public interface Set<T extends Comparable<T>>> {
  * Добавить ключ к множеству
  * Алгоритм должен быть как минимум lock-free
     * @param value значение ключа
     * @return false если value уже существует в множестве, true если элемент
был добавлен
     */
    boolean add(T value);
    /**
  * Удалить ключ из множества
  * Алгоритм должен быть как минимум lock-free
     * @param value значение ключа
     * @return false если ключ не был найден, true если ключ успешно удален
    boolean remove(T value);
    /**
  * Проверка наличия ключа в множестве
  * Алгоритм должен быть как минимум wait-free
     * @param value значение ключа
     * @return true если элемент содержится в множестве, иначе - false
     */
    boolean contains(T value);
    /**
  * Проверка множества на пустоту
  * Алгоритм должен быть как минимум lock-free
     * @return true если множество пусто, иначе - false
```

 $upaate: \\ 2019/11/15 \ courses: high_performance_computing: lock_free \ http://wiki.osll.ru/doku.php/courses: high_performance_computing: lock_free?rev=1573772230$ 01:57

```
*/
 boolean isEmpty();
 /**
* Возвращает lock-free итератор для множества
  * @return новый экземпляр итератор для множества
 java.util.Iterator<T> iterator();
```

Дополнительные условности:

- 1. Имя класса реализации SetImpl
- 2. Класс должен иметь конструктор без параметров
- 3. Pull Request должен содержать в части тестирования проходящие:
 - Нагрузочные тесты на основе jcstress
 - Тесты корректности на основе lincheck
- 4. В реализации не предполагается увидеть стандартные контейнеры из java.util.concurrent

From:

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

Permanent link:

http://wiki.osll.ru/doku.php/courses:high_performance_computing:lock_free?rev=1573772230

Last update: 2019/11/15 01:57



http://wiki.osll.ru/ Printed on 2025/11/05 11:17