Домашнее задание на 16 сентября, задача 3

Ульянин Дмитрий

14.09.2016

Пусть $k =$ `std::thread::hardware_concurrency`.

Тогда утверждается, что при $n > k$ `ticket_spinlock` будет работать неоптимально.

Действительно, пусть какой-то потом получил ticket и зашел в короткую критическую секцию. По условию поток может исполнить ее много раз за один квант времени, но вместо того, чтобы продолжить исполняться, ему выдается новый билет с номером, большим чем у других потоков. (Происходит это при $n > k$, т.к. иначе поток сразу сможет исполняться на свободном потоке ядра процессора).

При этом этот поток теперь будет ждать, пока не придет его очередь снова. Мы теряем время из-за того, что планировщик работает не очень быстро, ибо на это требуется несколько прерываний на процессоре и какие-то вычисления. А так же теряем время на то, что на каждый заход в критическую секцию каким-то процессом `ticket_spinlock` будет выполнять не очень тривиальные операции.

`test_and_set_spinlock` не подвержен этой проблеме, т.к. если процесс выполнил критическую секцию, но квант времени еще не израсходовал, то он просто войдет в `lock()` и `locked_.exchange(true)` вернет `false` и поток продолжит исполнять критические секции, пока квант не истечет.