Data Processing on Modern Hardware
Assignment 5 – Synchronization

Handout: **10^{th} June 2020**
Due: **17^{th} June 2020 by 9am**

Introduction

The goal of this exercise is to compare different synchronization apporaches introduced in this week's lecture. We analyze pessimistic and optimistic locking, exclusive and shared locks and vary the granularity from coarse- to fine-grained locking. As workload we use a list-based set provided in the code skeleton[1].

Part 1 - Implementation

Your task is synchronizing the given list-based set using the following approaches:

1. Coarse-Grained Locking
2. Coarse-Grained Locking with read/write lock
3. Lock Coupling
4. Lock Coupling with read/write locks
5. Optimistic
6. Optimistic Lock Coupling

Ignore memory reclamation (do not call delete).

Part 2 - Analysis and Evaluation

Apply the different synchronization techniques for the list-based set to analyze and compare them:

- How expensive is locking for the different approaches?
- How do the approaches perform in regard to #clock-cycles, #instructions and IPC?
- Which approach provides the best performance?

Vary the number of threads form low- to high contention, e.g., from 1-20 threads on the chair's SkylakeX machine. Further, investigate the effect of the domain parameter. Provide graphical plots for your results and normalize the results to make them comparable, e.g., using parallel speed-up as metric (lecture 7, slide 6).

[1] https://gitlab.db.in.tum.de/dpmh-ss20/hw5

Part 3 - In-depth analysis (voluntary)

This task is voluntary and does not count into the bonus. To get in-depth insides into the different synchronization approaches, analyze them using the Top-down microarchitecture analysis method (TMAM) and profile the code with the processor's hardware counters. Compare the different approaches for different numbers of threads in terms of:

- Costs per lock
- L1-cache-misses
- Blocked/idle CPU time

Again, normalize all results to make them comparable.

Submission guidelines

This homework has a duration of one week. Fork the repository, commit your changes in the git, and invite us (@*dpmh*) to hand in your homework.

The programming language of this homework is C++. We provide you a code skeleton, add functions needed for the implementation. For performance measuring of the experiments, you can either use the provided `perfEvent.hpp` and the commented code blocks or use the tools you applied in the previous assignments.