Студент групи К-25
Лазирко Ярослав Богданович

Варіант 6, кількість полів m=3 Частоти операцій для кожного поля: Поле 0: read — 20%, write — 5% Поле 1: read — 20%, write — 5% Поле 2: read — 20%, write — 5% Операція string: 25%

Для кожного з трьох полів використовується окремий std::shared_mutex, усього 3 м’ютекси. Операції read() використовують shared_lock, що дозволяє паралельне читання того самого поля.
Операції write() використовують unique_lock, блокуючи лише конкретне поле.Операція string() виконує консистентне читання всіх полів, послідовно захоплюючи всі три м’ютекси в режимі shared.
Оскільки у варіанті 6 значно переважають операції читання (60% читань та лише 15% записів), структура зі спільними м’ютексами на рівні кожного окремого поля забезпечує максимальний паралелізм.
Захоплення shared-блокування для string() не блокує читачів і лише короткочасно затримує писців, що є оптимальним у нашому розподілі навантаження.

	1 Thread	2 Thread	3 Thread
Variant 6	0.209423	0.463573	0.620593
Uniform	0.276121	0.614426	0.883509
Skewed	0.087205	0.185659	0.284081

У тесті Variant 6 структура даних працює найшвидше, що відповідає очікуванню, адже схема блокувань оптимізована під перевагу операцій читання. У тесті Uniform продуктивність знижується несуттєво, оскільки операції рівномірно розподілені між читанням, записом і string. У тесті Skewed спостерігається найгірший час виконання, оскільки 90% записів у поле 0 створюють високу конкуренцію за один і той самий м’ютекс, що збільшує чергу очікування потоків.
Такий результат повністю відповідає очікуванням.

У процесі роботи я самостійно зробив структуру даних з окремими м’ютексами на кожне поле. Частина роботи вимагала перевірити документацію Також писав генерацію файлів для трьох випадків і функцію, яка виконує операції в потоках. Вимірювання часу і запуск трьох потоків.