

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ
по лабораторной работе №2
по дисциплине «Параллельные алгоритмы»
Тема: Реализация взаимодействия потоков по шаблону «производитель-
потребитель»

Студент гр. 9303

Махаличев Н.А.

Преподаватель

Сергеева Е.И.

Санкт-Петербург

2022

Цель работы.

Изучение и практическое применение принципов синхронизации потоков на языке C++, реализация шаблона «производитель-потребитель».

Задание.

На базе лаб. 1 (части 1.2.1 и 1.2.2) реализовать итерационное (потенциально бесконечное) выполнение подготовки, обработки и вывода данных. Обеспечить параллельное выполнение потоков обработки готовой порции данных, подготовки следующей порции данных и вывода предыдущих полученных результатов.

Выполнение работы.

Для облегчения работы был произведён полный рефакторинг кода первой лабораторной работы, код структурирован, классифицирован и написан, опираясь на основы ООП.

Для решения проблемы производитель-потребитель написан класс Buffer, в котором определены два метода: `produce()` и `consume()`. Первый метод отвечает за добавление в очередь очередного набора данных (в данном случае объектов класса `Matrix`), в то время как второй извлекает необходимые данные из полученной очереди. Сам шаблон заключается в следующем:

1. Производитель добавляет в очередь данные;
2. Потребитель забирает полученные данные;
3. Если буфер пустой, потребитель ждёт поступления данных;
4. Если буфер полон, производитель ждёт его освобождения.

Данный шаблон реализуется с помощью синхронизации потоков. `Mutex` организывает взаимоисключающий доступ к данным, `lock` запрашивает у `mutex` монопольное использование общих данных, а `condition_variable` используется для блокировки потока до того момента, пока другой поток не изменит условие, используемое при блокировке, и не уведомит об этом `condition_variable`.

Программа разделена на три главных потока: генерация массивов, сложение массивов (выполняется с помощью параллельной работы N потоков) и вывод данных. Для решения поставленной задачи создаются три буфера, каждый из которых может иметь в очереди до восьми объектов класса Matrix: первые два буфера хранят первую и вторую матрицу, которые будут складываться, а третий хранит их сумму, которую необходимо вывести.

Демонстрация работы программы, а именно пятикратное сложение матриц 7x10 двумя потоками представлено на рис. 1.

| | | | | | | | | | | |
|----|------|------|------|------|------|------|------|------|------|------|
| 1 | -220 | -154 | 59 | 881 | 74 | -760 | 908 | 193 | -42 | -399 |
| 2 | 80 | 326 | -298 | -506 | -43 | 393 | 320 | -80 | 445 | -557 |
| 3 | 60 | 13 | 25 | 0 | 51 | 697 | 143 | -20 | 564 | 152 |
| 4 | -352 | 346 | -2 | -292 | -69 | 424 | -51 | 191 | -31 | -389 |
| 5 | 145 | 401 | 289 | 199 | 599 | -48 | -55 | -81 | -128 | 94 |
| 6 | 715 | -68 | 459 | -556 | 285 | -138 | 142 | -867 | 548 | -293 |
| 7 | 285 | 196 | -243 | -12 | 256 | -608 | 412 | -91 | 584 | 86 |
| 8 | | | | | | | | | | |
| 9 | -776 | -886 | 368 | 88 | -172 | 313 | 879 | 212 | -340 | 93 |
| 10 | 103 | -690 | -212 | -563 | -835 | 426 | 256 | -521 | -601 | 269 |
| 11 | -836 | 292 | -30 | -203 | 242 | -506 | 245 | 234 | 338 | 329 |
| 12 | -98 | -86 | -204 | -25 | 354 | -376 | -360 | 587 | 188 | 5 |
| 13 | 680 | -356 | 19 | -180 | 433 | -464 | -401 | -310 | 16 | -1 |
| 14 | 663 | -116 | -4 | -366 | 34 | 239 | 481 | 280 | 178 | 523 |
| 15 | -686 | -215 | -858 | -890 | 113 | 496 | 87 | 457 | -565 | -373 |
| 16 | | | | | | | | | | |
| 17 | -426 | -252 | 462 | 184 | -224 | 570 | 305 | -479 | -735 | 418 |
| 18 | 261 | -271 | 400 | 14 | 152 | 391 | 893 | -55 | -54 | -665 |
| 19 | -83 | 499 | -32 | -117 | 225 | 120 | -620 | -533 | -743 | -219 |
| 20 | 170 | -168 | 530 | 337 | -632 | -342 | 259 | 25 | 531 | -475 |
| 21 | -203 | 496 | -42 | 197 | -784 | 462 | -707 | 109 | -240 | -408 |
| 22 | 148 | 678 | -557 | 469 | 265 | 20 | -59 | -355 | 839 | 199 |
| 23 | -574 | 363 | 383 | -340 | -300 | 104 | 24 | 312 | 481 | 907 |
| 24 | | | | | | | | | | |
| 25 | -11 | 327 | -6 | -299 | -644 | 707 | -183 | 162 | -288 | -662 |
| 26 | 97 | -449 | 108 | -414 | 292 | 217 | 52 | 285 | -233 | 109 |
| 27 | 543 | 945 | 74 | -390 | 383 | -295 | -608 | -36 | 270 | -795 |
| 28 | -134 | -389 | -762 | -140 | 664 | -54 | -80 | -167 | -540 | 336 |
| 29 | 523 | -443 | 240 | 336 | -153 | -116 | -95 | -101 | 521 | 24 |
| 30 | -287 | 417 | -325 | -212 | -621 | -590 | -155 | 123 | -273 | 467 |
| 31 | 34 | 945 | -218 | 624 | 511 | 446 | -78 | -865 | 633 | -266 |
| 32 | | | | | | | | | | |
| 33 | 43 | -637 | -496 | 479 | -516 | -533 | 569 | -662 | 109 | -137 |
| 34 | -8 | 154 | -171 | -409 | 254 | -46 | 56 | -728 | -611 | -690 |
| 35 | -10 | 767 | 494 | 290 | -51 | -99 | -470 | 540 | 78 | 420 |
| 36 | 645 | 473 | 136 | 501 | -695 | -27 | -327 | -774 | -336 | -218 |
| 37 | 89 | 8 | -711 | 271 | -49 | 543 | -71 | 360 | -185 | -329 |
| 38 | -625 | -490 | -210 | -427 | 152 | -260 | -526 | 34 | 633 | 904 |
| 39 | 159 | -370 | 82 | 0 | -165 | -613 | 325 | 509 | 317 | 341 |

Рисунок 1 — Результат работы программы.

Выводы.

В процессе выполнения лабораторной работы были изучены и применены на практике принципы синхронизации потоков на языке C++ с помощью lock, mutex и condition_variable. Был написан код реализации шаблона «производитель-потребитель».