Отчет по задаче GameOfLife – Новогородцев Степан

Постановка задачи:

Реализовать простейшую модель игры **John Conway's Game of Life,** применив полученные навыки из прослушанных лекций по курсу Многопоточное программирование и исследовать результаты, полученные при использовании многопоточности.

Реализация:

В работе представлены 2 реализации предложенного интерфейса GameOfLife – однопоточная и многопоточная. Все тесты для файлов “input”,”input100”,”input1000” проходят успешно.

Однопоточная модель – в одном потоке считывается матрица nxn и для каждой ячейки считается число соседей и по правилам Conway’s Game of Life происходит эволюция.

Многопоточная модель – происходит разбиение по сегментам общей матрицы в классе MainFrame. Дальше на каждый сегмент запускается свой поток и после того, как клетка эволюционировала (метод doSmth()), выставляется флаг об окончании итерации. Когда все потоки выполняют работу, создается новая матрица шага t, после чего эти шаги повторяются.

Результаты:

Получены результаты обработки 3 файлов для однопоточной и многопоточной реализации.

Для простоты изложения, обозначим тесты для файлов input (9x9, 1 поколение), как “test1”, input100 (100x100 10 поколений) , как “test100” и input1000 (1000x1000, 100 поколений), как “test1000”.

Однопоточная модель:

Время выполнения тестов:

Test1 ~ 100ms

Test100 ~ 40000ms

Test1000 ~ 7,4s

Многопоточная модель:

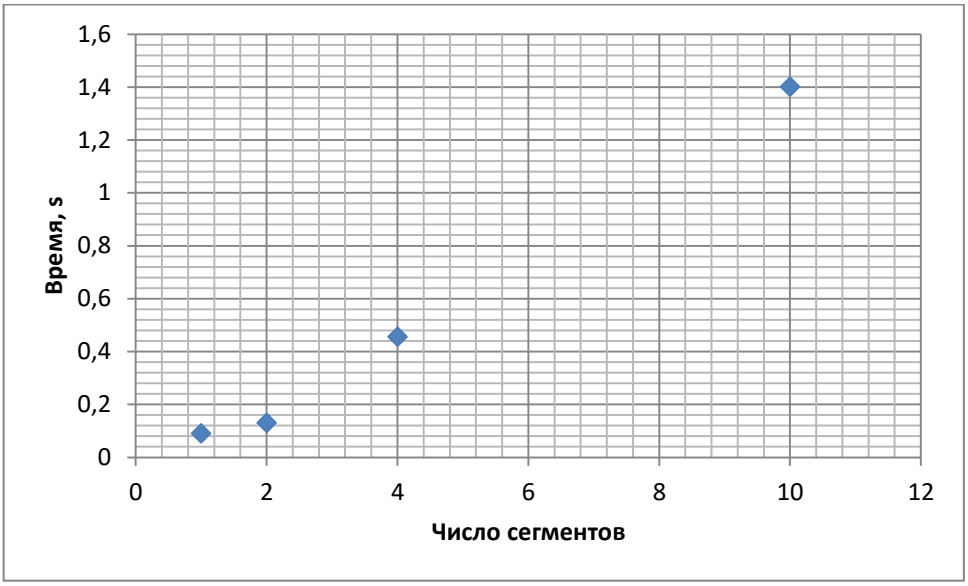
Для многопоточной модели выбирается число разбиений на сегменты (пример – n=1 – строка, n=2 – 2 прямоугольника высотой N/2, n=N – вся матрица)

Test 1

Число сегментов	1	3	9
Время, ms	~ 3900	~ 50000	~350000

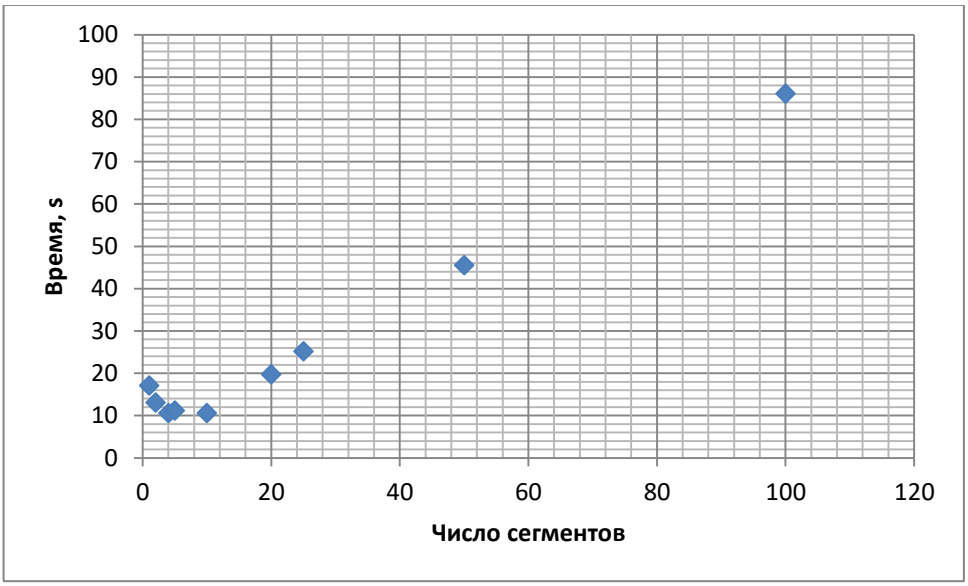
Test100

Число сегментов	1	2	4	10	100
Время, s	~0,09	~0,13	~0,455	~1,4	~10



Test1000

Число сегментов	1	2	4	5	10	20	25	50	100
Время, s	17	13	10,5	11,1	10,5	19,7	25,1	45,5	86



Из полученных результатов видно, что для данной реализации, однопоточная модель отрабатывает быстрее. Для многопоточной реализации графики показывают, что наличие большого числа тредов является не самым лучшим способом оптимизации. Из чего можно сделать выводы, для данной реализации слишком много времени уходит на синхронизацию потоков.

Возможная оптимизация:

В ходе работы у меня появились идеи о неплохой, на мой взгляд, реализации: можно выделить несколько потоков (~2-8), которые будут наблюдать за живыми клетками. Начальные данные – список живых клеток, они в свою очередь могут достать информацию о своих соседях (8 или более), но что, если создать отдельную матрицу из просмотренных/непросмотренных соседях? Таким образом уменьшается время на проверку, но возникает вопрос – что делать если такие зоны будут пересекаться? Над реализацией данной идеи я потратил какую-то часть времени, которая не увенчалась успехом. Представлены проверенные реализации.