Отчёт по лабораторной работе №1

Жуков Вадим, ИВТ-12М

Технические характеристики

OС: Windows 10

VS: Visual Studio 2017 v15.9.18

IPS: v2019

Железо: 2 физических ядра, 4 логических, 3.4ГГц

# Задание 1

Создайте пустой проект C++ в VS. Добавьте, напишите, отладьте исходные коды для расчета интеграла по Вашему варианту. Оцените время и точность (относительно аналитического значения) расчета интеграла в зависимости от количества интервалов (равномерное разбиение, 100, 1000, 10000, 100000, 1000000).

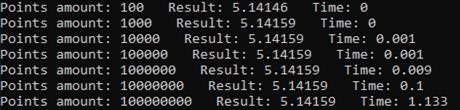
Результаты работы приведены на рисунке 1.

Рисунок 1. Задание 1, результат

В первых 2 случаях время равно нулю, потому что оно получилось слишком маленьким, и я количество знаков после запятой я оставил стандартным.

# Задание 2

Создайте новый проект. С использованием потоков (thread, mutex), автоматической параллелизацией(/Qpar), автоматической векторизацией (отключение векторизации для сравнения) напишите программу, для решения Вашей задачи. Оцените точность и время выполнения программы, запуская ее с теми же параметрами, что и последовательную программу. Есть ли выигрыш по времени выполнения?

Доп. материал:

<https://docs.microsoft.com/ru-ru/cpp/parallel/parallel-programming-in-visual-cpp?view=vs-2019>

Результаты показаны на рисунке 2. Как видно по скриншоту, вычисление с помощью потоков работает эффективнее, но на более большом объёме данных (в самом последнем случае). Скорее всего это связано с выполнением операций по обслуживанию потоков. На самом деле, в данном задании сложно дать действительно объективный объект, потому что в институте поведение было немного другим.

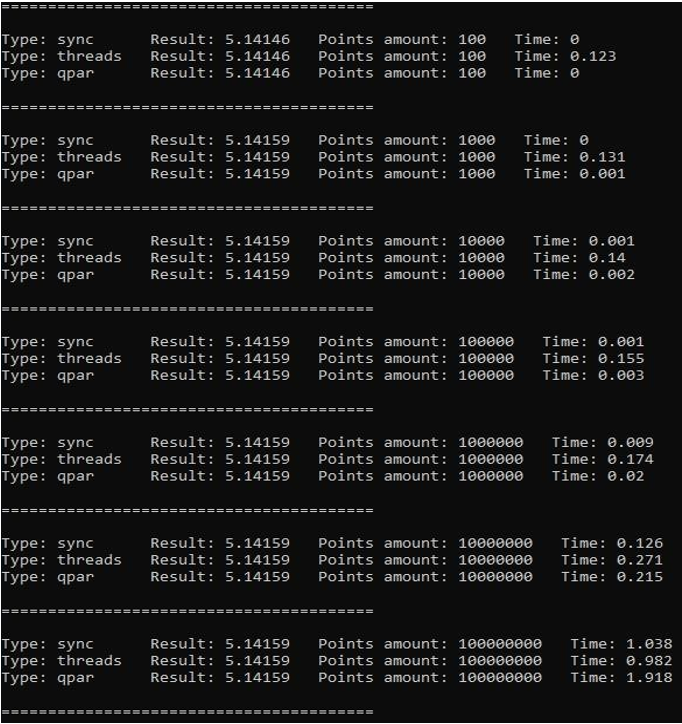


Рисунок 2. Задание 2, результат

# Задание 3

Воспользоваться VTune и Inspector для последовательной программы.  Добавить cilk\_for, проверить работоспособность, правильность решения. Воспользоваться VTune и Inspector и убедиться в отсутствии ошибок и гонок. Добавить reducer, проанализировать новое решение и убедиться в корректности его работы.

Результаты работы с cilk\_for представлены на рисунке 3. Гонок данных не обнаружено, что видно на рисунке 4. Также анализ программы с помощью ***Intel VTune Amplifier XE*** представлен нарисунке 5 и рисунке 6***.***

Как видно, вычисления с помощью cilk\_for заняли больше времени, чем даже простое последовательное вычисление из задания 1. Это связано с различными обслуживающими операциями, которые подъедают время выполнения, и с тем, что данная задача не является достаточно “увесистой”, чтобы её распараллеливать.

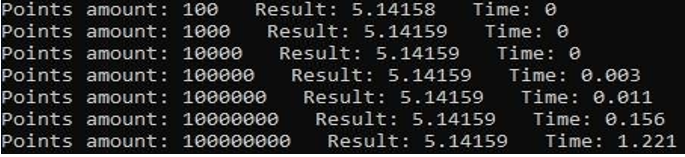


Рисунок 3. Задание 3, результат

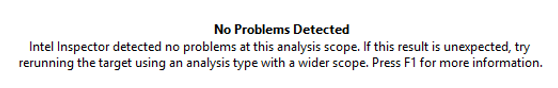


Рисунок 4. Задание 3, гонки данных

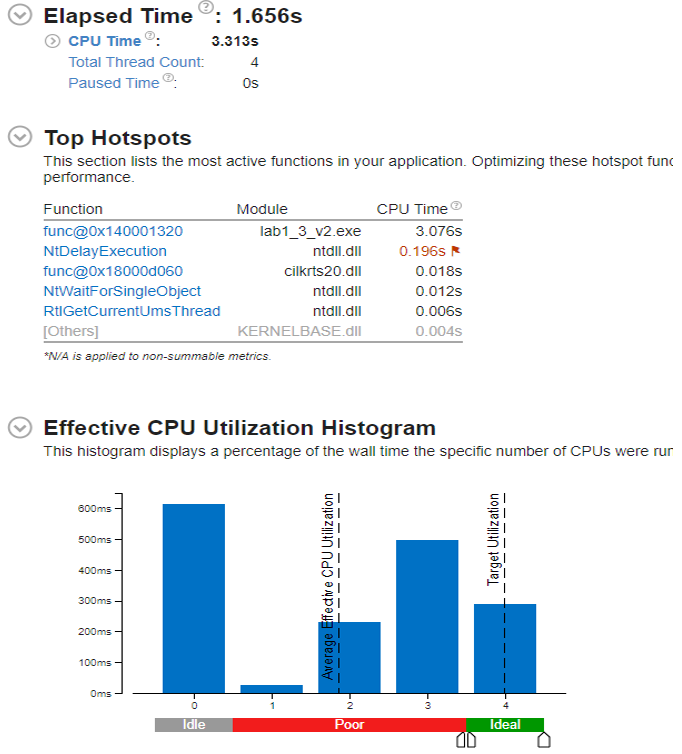


Рисунок 5. Задание 3, VTune 1

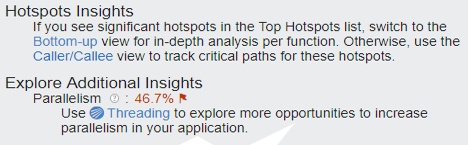


Рисунок 6. Задание 3, VTune 2

# Задание 4

Задание 3, но с использованием TBB.

Результаты представлены на рисунке 7. Программа стала работать ещё медленнее…

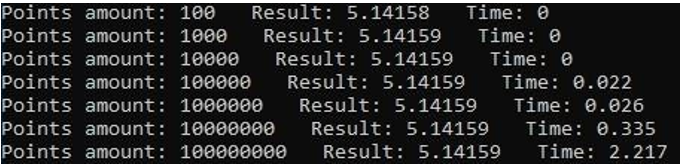


Рисунок 7. Задание 4, результат

В целом, совокупно по всем результатам видно, что инструмент работает, но им надо уметь пользоваться и применять его для более специфичных задач. С вычислением интеграла это, видимо, не работает, потому что много времени занимает работа с описанными инструментами и объём данных не настолько большой, чтобы программу распараллеливать.