Отчёт по лабораторной работе № 1
Жуков Вадим, ИВТ-12М
Вариант 9

Интеграл: $\int_{-1}^{1} \frac{4}{(1+x^2)^2} dx$

Способ решения: метод трапеций

Аналитическое решение: $\frac{2x}{x^2+1} + 2\text{atan}(x)$

Ответ: $2 + \pi$

1. Последовательная программа по расчету интеграла (1 занятие)

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).

```
Points amount: 100   Result: 5.14146   Time: 0
Points amount: 1000   Result: 5.14159   Time: 0
Points amount: 10000   Result: 5.14159   Time: 0.001
Points amount: 100000   Result: 5.14159   Time: 0.001
Points amount: 1000000   Result: 5.14159   Time: 0.009
Points amount: 10000000   Result: 5.14159   Time: 0.1
Points amount: 100000000   Result: 5.14159   Time: 1.133
```

2. Программа по расчету интеграла с использованием нескольких потоков и векторных инструкций (1 занятие)

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

```
========================================

Type: sync      Result: 5.14146   Points amount: 100   Time: 0
Type: threads   Result: 5.14146   Points amount: 100   Time: 0.123
Type: qpar      Result: 5.14146   Points amount: 100   Time: 0

========================================

Type: sync      Result: 5.14159   Points amount: 1000   Time: 0
Type: threads   Result: 5.14159   Points amount: 1000   Time: 0.131
Type: qpar      Result: 5.14159   Points amount: 1000   Time: 0.001

========================================

Type: sync      Result: 5.14159   Points amount: 10000   Time: 0.001
Type: threads   Result: 5.14159   Points amount: 10000   Time: 0.14
Type: qpar      Result: 5.14159   Points amount: 10000   Time: 0.002

========================================

Type: sync      Result: 5.14159   Points amount: 100000   Time: 0.001
Type: threads   Result: 5.14159   Points amount: 100000   Time: 0.155
Type: qpar      Result: 5.14159   Points amount: 100000   Time: 0.003

========================================

Type: sync      Result: 5.14159   Points amount: 1000000   Time: 0.009
Type: threads   Result: 5.14159   Points amount: 1000000   Time: 0.174
Type: qpar      Result: 5.14159   Points amount: 1000000   Time: 0.02

========================================

Type: sync      Result: 5.14159   Points amount: 10000000   Time: 0.126
Type: threads   Result: 5.14159   Points amount: 10000000   Time: 0.271
Type: qpar      Result: 5.14159   Points amount: 10000000   Time: 0.215

========================================

Type: sync      Result: 5.14159   Points amount: 100000000   Time: 1.038
Type: threads   Result: 5.14159   Points amount: 100000000   Time: 0.982
Type: qpar      Result: 5.14159   Points amount: 100000000   Time: 1.918

========================================
```

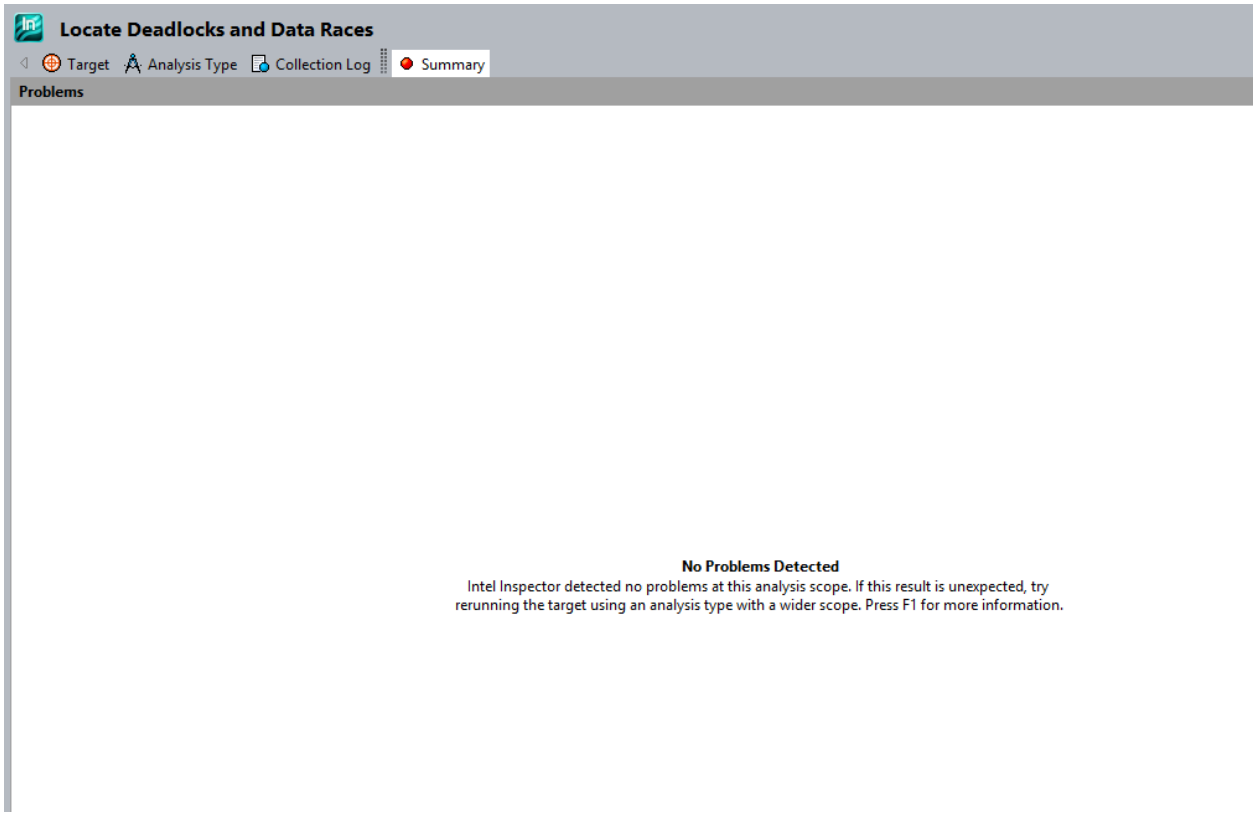
В институте qpar и потоки выигрывали по времени в 2 раза, на домашнем компьютере такого результата не получилось. Перепробовав различные вариации по количеству потоков удовлетворительного результата так и не получил. К сожалению, не знаю, как решить эту проблему.

3. Программа с использованием дополнений Intel Cilk Plus языка C++ (2 занятие)

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.

```
Points amount: 100   Result: 5.14158   Time: 0
Points amount: 1000   Result: 5.14159   Time: 0
Points amount: 10000   Result: 5.14159   Time: 0
Points amount: 100000   Result: 5.14159   Time: 0.003
Points amount: 1000000   Result: 5.14159   Time: 0.011
Points amount: 10000000   Result: 5.14159   Time: 0.156
Points amount: 100000000   Result: 5.14159   Time: 1.221
```

Проверка с помощью ***Intel Parallel Inspector XE:***



Оценка эффективности с помощью ***Intel VTune Amplifier XE:***

Elapsed Time: 1.656s

CPU Time:	3.313s
Total Thread Count:	4
Paused Time:	0s

Top Hotspots

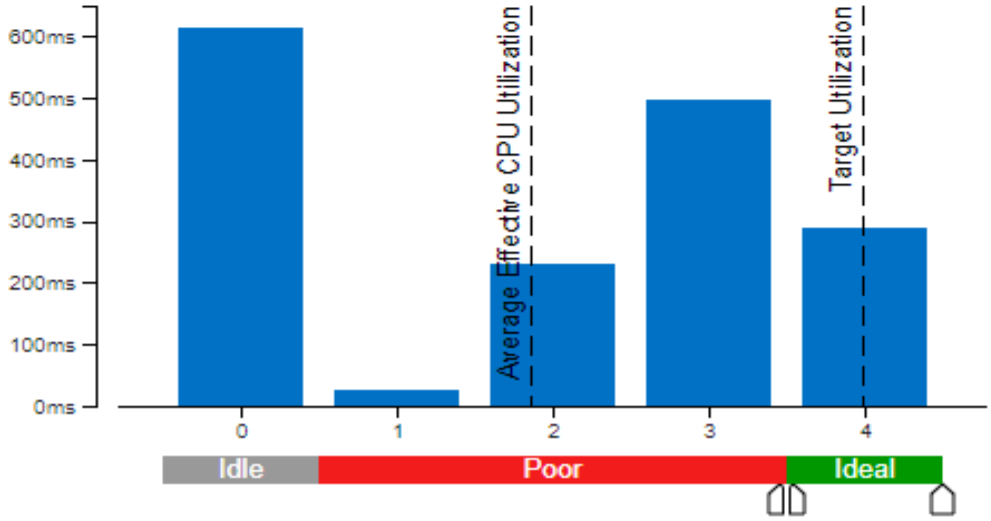
This section lists the most active functions in your application. Optimizing these hotspot func performance.

Function	Module	CPU Time
func@0x140001320	lab1_3_v2.exe	3.076s
NtDelayExecution	ntdll.dll	0.196s
func@0x18000d060	cilkrts20.dll	0.018s
NtWaitForSingleObject	ntdll.dll	0.012s
RtlGetCurrentUmsThread	ntdll.dll	0.006s
[Others]	KERNELBASE.dll	0.004s

**N/A is applied to non-summable metrics.*

Effective CPU Utilization Histogram

This histogram displays a percentage of the wall time the specific number of CPUs were rur



Hotspots Insights

If you see significant hotspots in the Top Hotspots list, switch to the Bottom-up view for in-depth analysis per function. Otherwise, use the Caller/Callee view to track critical paths for these hotspots.

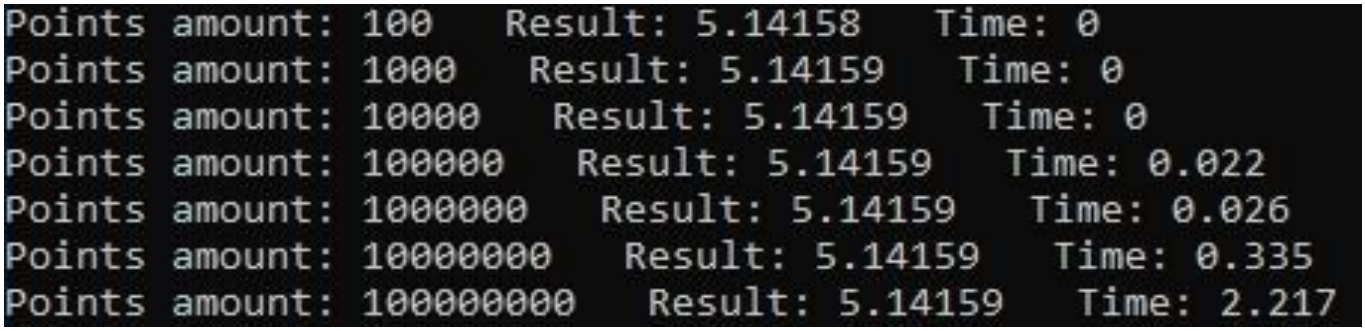
Explore Additional Insights

Parallelism : 46.7%

Use Threading to explore more opportunities to increase parallelism in your application.

4. Программа с использованием шаблонов TBB (3 занятие)

Коды программ необходимо загрузить на ***Github (проектами)***. По результатам работы должен быть написан отчет, отражающий все этапы выполнения задания. После написания всех программ необходимо сделать отчет.



Сведения о зависимости времени выполнения от заданных параметров алгоритма:
задаваемые параметры алгоритма – это границы интегрирования и точность (количество точек разбиения). Очевидно, что чем выше точность, тем больше время выполнения. При распараллеливании программы время должно уменьшаться в зависимости от количества потоков (например, при выполнении программы 2 потоками время уменьшалось в 2 раза), но выполняется это не на всех компьютерах и имеет смысл только при действительно большом количестве вычислений.