

Индивидуальное задание.

Напишите параллельную программу вычисления следующего интеграла с использованием дополнений Intel Cilk Plus языка C++:

$$\int_0^1 \frac{4}{\sqrt{4-x^2}} dx$$

1. Описание проблемы и краткая характеристика инструментов параллелизации, используемых для решения задачи

Посчитаем интеграл аналитически. Интеграл табличный, поэтому сделать это не составит труда.

$$\int_0^1 \frac{4}{\sqrt{2^2-x^2}} dx = 4 \arcsin\left(\frac{1}{2}\right) - 4 \arcsin\left(\frac{0}{2}\right) = \frac{4\pi}{6} = \frac{2\pi}{3} \cong 2,093(3)$$

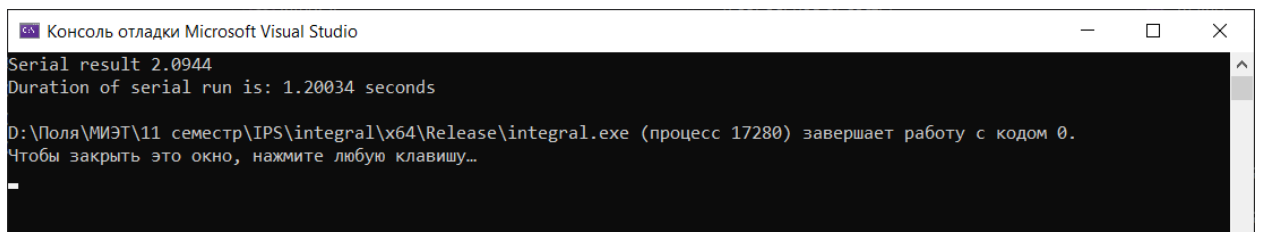
Далее в работе будем рассматривать численные методы нахождения интеграла. Проще всего использовать метод прямоугольников.

Для проверки реализуем последовательное вычисление интеграла.

```
double serial_integral_rect(std::function<double(double)> fun, double a, double b, size_t
num_h) {
    double sum(0.);
    double h = (b - a) / static_cast<double>(num_h);
    size_t num_batch = 1024;

    for (size_t i = 0; i < num_h; i++){//+=num_batch) {
        sum += fun(a + h * i) + fun(a + h * (i + 1));
    }

    return sum * h / 2.;
}
```



Консоль отладки Microsoft Visual Studio

```
Serial result 2.0944
Duration of serial run is: 1.20034 seconds

D:\Поля\МИЭТ\11 семестр\IPS\integral\x64\Release\integral.exe (процесс 17280) завершает работу с кодом 0.
Чтобы закрыть это окно, нажмите любую клавишу...
```

Работает корректно.

Инструменты параллелизации:

— Intel Cilk Plus — расширение языка Си++, призванное упростить написание многопоточных программ. Cilk Plus представляет собой динамический планировщик исполнения потоков и набор ключевых слов, сообщающих компилятору о возможности применения той или иной схемы планирования.

— Intel Parallel Inspector — инструмент для обнаружения ошибок памяти и потоков в последовательных и параллельных приложениях на платформах Windows и Linux.

— Intel VTune Amplifier — это средство для оптимизации производительности и профилировки параллельных приложений

2. Описание и анализ программной реализации

Определим наиболее часто используемые участки кода с помощью *Intel VTune Amplifier XE*.

HotspotsHotspots by CPU Utilization

Analysis ConfigurationCollection LogSummaryBottom-upCaller/CalleeTop-down TreePlatform

Elapsed Time: 1.034s

CPU Time: 0.967s

Total Thread Count: 1

Paused Time: 0s

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions typically results in improving overall application performance.

Function	Module	CPU Time
func	integral.exe	0.488s
main	integral.exe	0.274s
std::_Func_impl_no_alloc<double (*)(double),double,double>::_Do_call	integral.exe	0.189s
exit	ucrtbas e.dll	0.016s

*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 running simultaneously. Spin and Overhead time adds to the Idle CPU utilization value.

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: 23.4%

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

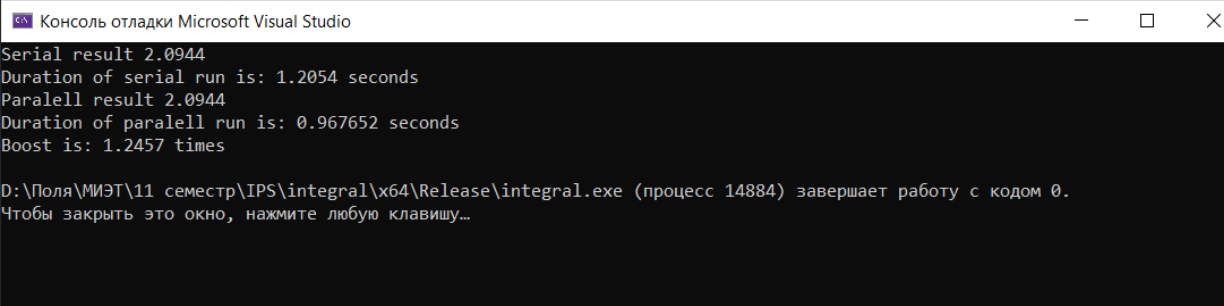
INSIGHTS

Реализуем параллельный метод прямоугольников.

```
double paralell_integral_rect(std::function<double(double)> fun, double a, double b,
size_t num_h) {
    cilk::reducer_opadd<double> sum(0.);
    double h = (b - a) / static_cast<double>(num_h);
    size_t num_batch = 1024;

    cilk_for (size_t i = 0; i < num_h; i++){// += num_batch) {
        sum += fun(a + h * i) + fun(a + h * (i + 1));
    }

    return sum->get_value() * h / 2.;
}
```



По данным результатам видно, что а) метод работает правильно и б) метод работает быстрее.

Оценим параллельную реализацию с помощью Intel VTune Amplifier XE.

HotspotsHotspots by CPU Utilization

Analysis ConfigurationCollection LogSummaryBottom-upCaller/CalleeTop-down TreePlatform

Elapsed Time: 1.785s

CPU Time: 3.308s

Effective Time: 2.775s

Spin Time: 0.034s

Overhead Time: 0.499s

Total Thread Count: 4

Paused Time: 0s

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions typically results in improving overall application performance.

Function	Module	CPU Time
func	integral.exe	1.625s
func@0x1400014c0	integral.exe	0.868s
[Cilk reducer]	cilkrt20.dll	0.351s
std::_Func_impl_no_alloc<double (*)(double),double,double>::_Do_call	integral.exe	0.268s
_cilkrt2_get_tls_worker	cilkrt20.dll	0.149s
[Others]		0.048s

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

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

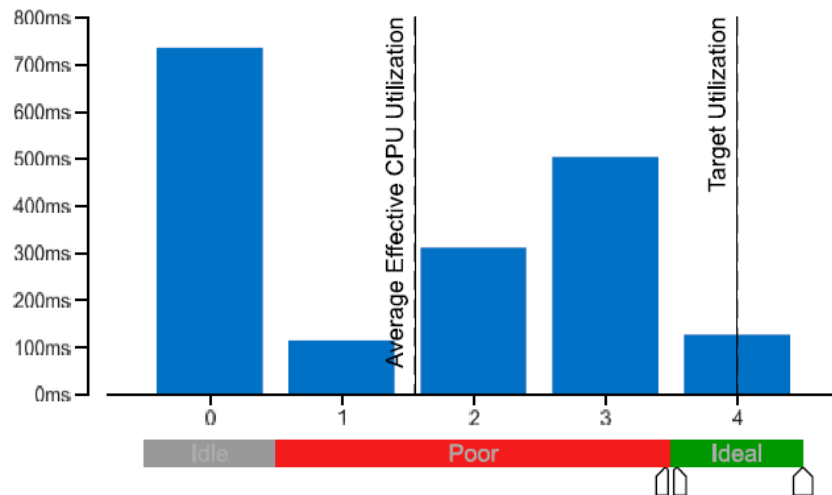
Parallelism: 38.9%

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

INSIGHTS

Effective CPU Utilization Histogram

This histogram displays a percentage of the wall time the specific number of CPUs were running simultaneously. Spin and Overhead time adds to the Idle CPU utilization value.



Проверим программу с помощью инструмента *Intel Parallel Inspector XE*.

The screenshot shows the Intel Parallel Inspector XE interface. The top bar displays 'Detect Deadlocks and Data Races'. The 'Problems' pane on the left shows a single issue: 'Data race' with ID 'P1', source '[Unknown]; reducer_opadd.h', and module 'integral.exe'. The 'Filters' pane on the right shows the issue details: Severity (Error), Type (Data race), Source ([Unknown], reducer_opadd.h), and Module (integral.exe). The 'Code Locations: Data race' pane at the bottom shows two write operations to memory locations 'integral.exe!0x15e3' and 'integral.exe!0x15d2'. The 'Timeline' pane on the right shows the execution flow with 'main (12668)' and 'Cilk Worker (5744)'.

ID	Type	Sources	Modules	State
P1	Data race	[Unknown]; reducer_opadd.h	integral.exe	New

Description	Source	Function	Module	Variable
Write	integral.exe:0x15e3	[Unknown]	integral.exe	0x1df558f7100
Symbol information not found. Suggestion: Specify locations in a Project Properties dialog box search tab, then re-resolve the result.				
Write	integral.exe:0x15d2	[Unknown]	integral.exe	0x1df558f7100
Symbol information not found. Suggestion: Specify locations in a Project Properties dialog box search tab, then re-resolve the result.				

Intel Inspector

Detect Memory Problems

Target | Analysis Type | Collection Log | Summary

ID	Type	Sources	Modules	Object Size	State
P1	Mismatched allocation/dealloca...	exe_common.inl; xloci...	integral.exe		New
P2	Mismatched allocation/dealloca...	[Unknown]; integral.c...	integral.exe; ucrtbas...		New
P3	Mismatched allocation/dealloca...	[Unknown]	ucrtbase.dll		New
P4	Mismatched allocation/dealloca...	[Unknown]; xlocinfo	integral.exe; ucrtbas...		New
P5	Mismatched allocation/dealloca...	xlocinfo	integral.exe		New
P6	Mismatched allocation/dealloca...	xlocinfo	integral.exe		New
P7	Memory not deallocated	streambuf; xlocinfo	integral.exe	4440	New

Filters

Severity

- Error: 6 item(s)
- Warning: 1 item(s)

Type

- Memory not deallocated: 1 item(s)
- Mismatched allocation/deallocati...: 6 item(s)

Source

- [Unknown]: 3 item(s)
- exe_common.inl: 1 item(s)

Code Locations: Mismatched allocation/deallocation

Description	Source	Function	Module	Object Size	Offset	Variable
Mismatched dealloc...	integral.cpp:61	main	integral.exe			block
<pre> 59 std::chrono::high_resolution_clock::t 60 61 std::cout << "Serial result " << resu 62 std::chrono::duration<double> duratio 63 std::cout << "Duration of serial run </pre>						
Allocation site	ucrtbase.dll:0xae8	calloc_base	ucrtbase.dll			block
<p>Symbol information not found. Suggestion: Specify locations in a Project Properties dialog box search tab, then re-resolve the result</p> <pre> ucrtbase.dll!calloc_base ucrtbase.dll!setlocale ucrtbase.dll!setlocale </pre>						

Timeline

rtlAnsiStringToUnicodeString (17212)

Оценим зависимость времени выполнения от заданных параметров алгоритма.

Консоль отладки Microsoft Visual Studio

```

Serial result 2.0944
Duration of serial run is: 7e-06 seconds
Paralell result 2.0944
Duration of paralell run is: 0.0005136 seconds
Boost is: 0.0136293 times

Serial result 2.0944
Duration of serial run is: 1.25e-05 seconds
Paralell result 2.0944
Duration of paralell run is: 7.3e-05 seconds
Boost is: 0.171233 times

Serial result 2.0944
Duration of serial run is: 0.00014 seconds
Paralell result 2.0944
Duration of paralell run is: 0.0002178 seconds
Boost is: 0.642792 times

Serial result 2.0944
Duration of serial run is: 0.0012736 seconds
Paralell result 2.0944
Duration of paralell run is: 0.0015661 seconds
Boost is: 0.81323 times

Serial result 2.0944
Duration of serial run is: 0.0141216 seconds
Paralell result 2.0944
Duration of paralell run is: 0.0129008 seconds
Boost is: 1.09463 times

D:\Поля\МИЭТ\11 семестр\IPS\integral\x64\Release\integral.exe (процесс 13124) завершает работу с кодом 0.
Чтобы закрыть это окно, нажмите любую клавишу...

```

Видим, что при увеличении количества точек разбиения отрезка интегрирования (n), параллельный метод начинает работать быстрее.