Индивидуальное задание

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

$$\int_0^1 \frac{8}{1+x^2}\,dx.$$

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

В математическом анализе обосновывается аналитический способ нахождения значения интеграла с помощью формулы Ньютона-Лейбница

$$\int_a^b f(x)dx = F(b) - F(a)$$

Однако применение данного подхода к вычислению наталкивается на несколько серьезных препятствий:

- Для многих функций не существует первообразной среди элементарных функций;
- Даже если первообразная для заданной функции существует, то вычисление двух ее значений $F(a)$, $F(b)$ может оказаться более трудоемким, чем вычисление существенно большего количества значений $f(x)$;
- Для многих реальных приложений определенного интеграла характерная дискретность задания подынтегральной функции, что делает аналитический подход неприменимым.

Сказанное предопределяет необходимость использования приближенных формул для вычисления определенного интеграла на основе значений подынтегральной функции. Такие специальные формулы называются квадратурными формулами или формулами численного интегрирования.

В качестве инструментов параллелизации, используемых для решения задачи, мы будем использовать:

- Cilk Plus – это расширения языка C/C++, которое помогает с введением параллелизма в код программы;
- Intel Parallel Studio XE – это набор программных продуктов от компании Intel;

1. Intel Parallel Inspector – инструмент, предназначенный для тестирования работающей программы с целью выявления основных ошибок, которые возникают при разработке параллельного кода;
2. Intel Amplifier – инструмент, который используется для профилирования приложения с целью выявления наиболее часто используемых участков программы (hotspots), а также узких мест (bottleneck) в работе программы. Этот инструмент также позволяет анализировать параллельные программы на эффективность использования ими ресурсов процессора;

Описание и анализ программной реализации.

Реализуем вычисление интеграла с помощью численного интегрирования методом правых прямоугольников.

```
double integrate(double a, double b, size_t N)
{
	const double h = (b - a) / N;
	double sum = 0.0f;
	for (size_t i = 0; i < N; ++i)
		sum += fun(a + i * h);

	return sum * h;

}
```

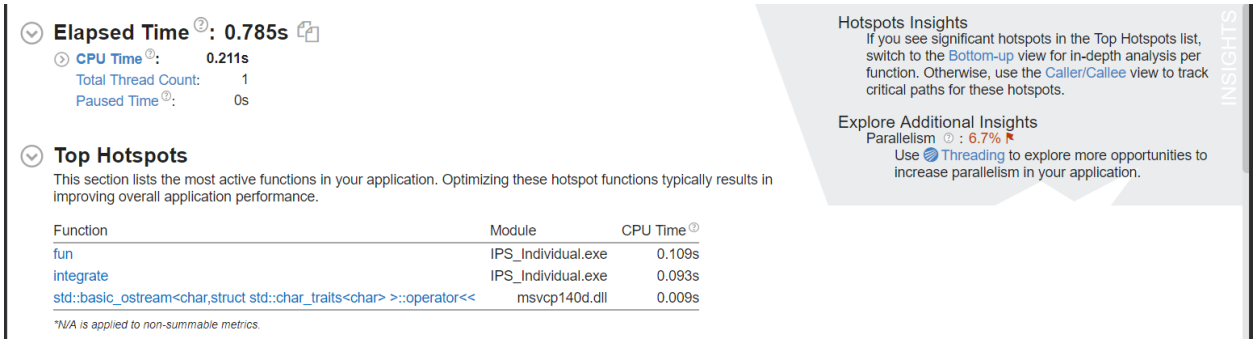
Аналитическое значение интеграла $\int_0^1 \frac{8}{1+x^2} dx = 2\pi$.

Запусти программу, получим численный результат

```
Non parallel ans: 6.28319
Estimated time: 0.214222
```

Численный результат приблизительно равен 2π.

С помощью инструмента Amplifier XE определим наиболее часто используемые участки кода.

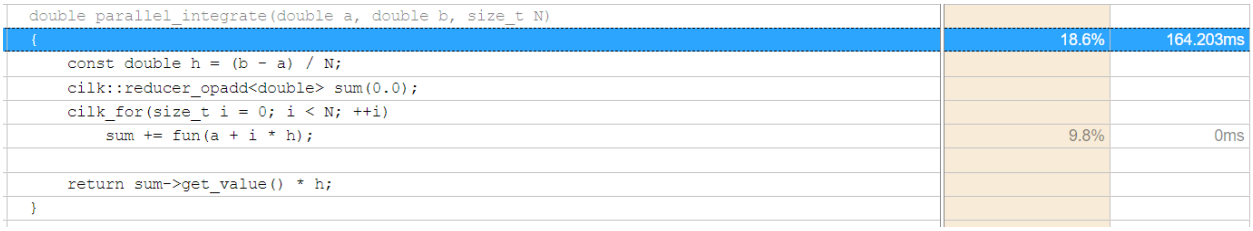


Видим, что большую часть времени вычисляется сумма. Этого и следовало ожидать. Распараллелим вычисление суммы с помощью *cilc :: reducer_opadd*.

Параллельная версия:

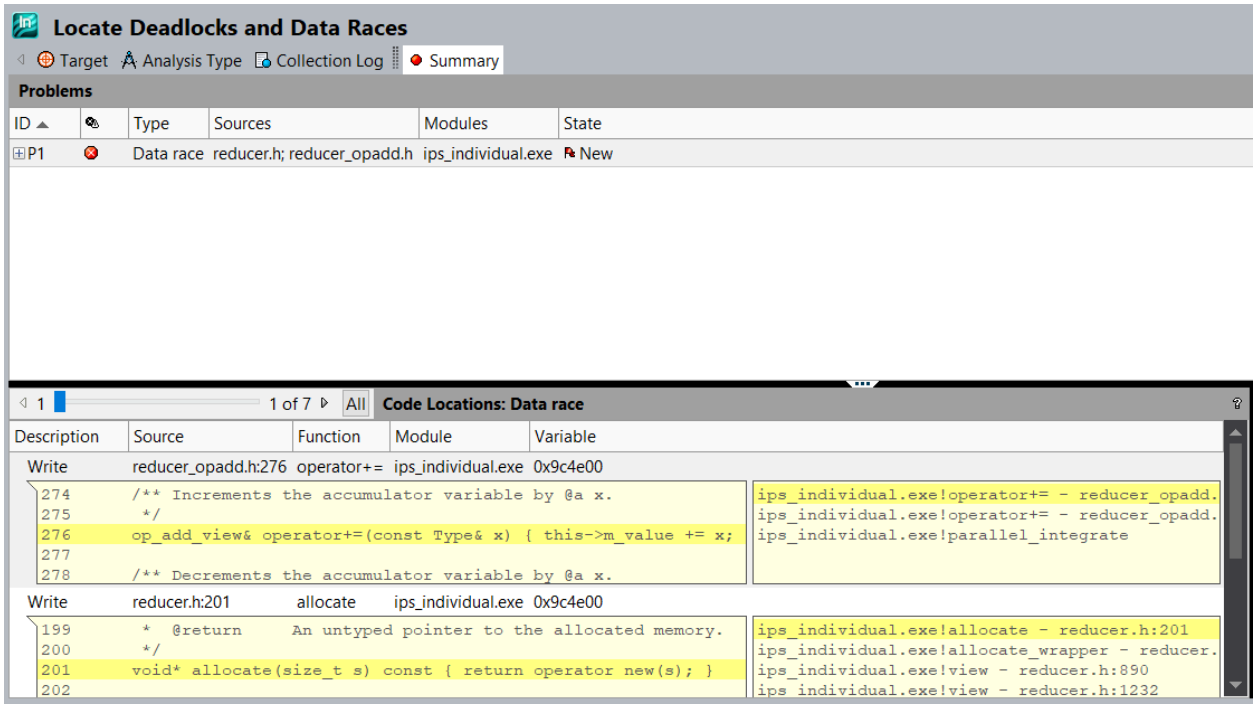
```
double parallel_integrate(double a, double b, size_t N)
{
	const double h = (b - a) / N;
	cilk::reducer_opadd<double> sum(0.0);
	cilk_for(size_t i = 0; i < N; ++i)
		sum += fun(a + i * h);

	return sum->get_value() * h;

}
```

Запустим Amplifier XE для параллельной версии:



Видим, что стало лучше.

Далее, используя Inspector XE, определим те данные, которые принимают участие в гонке данных или в других основных ошибках, возникающих при разработке параллельных программ.

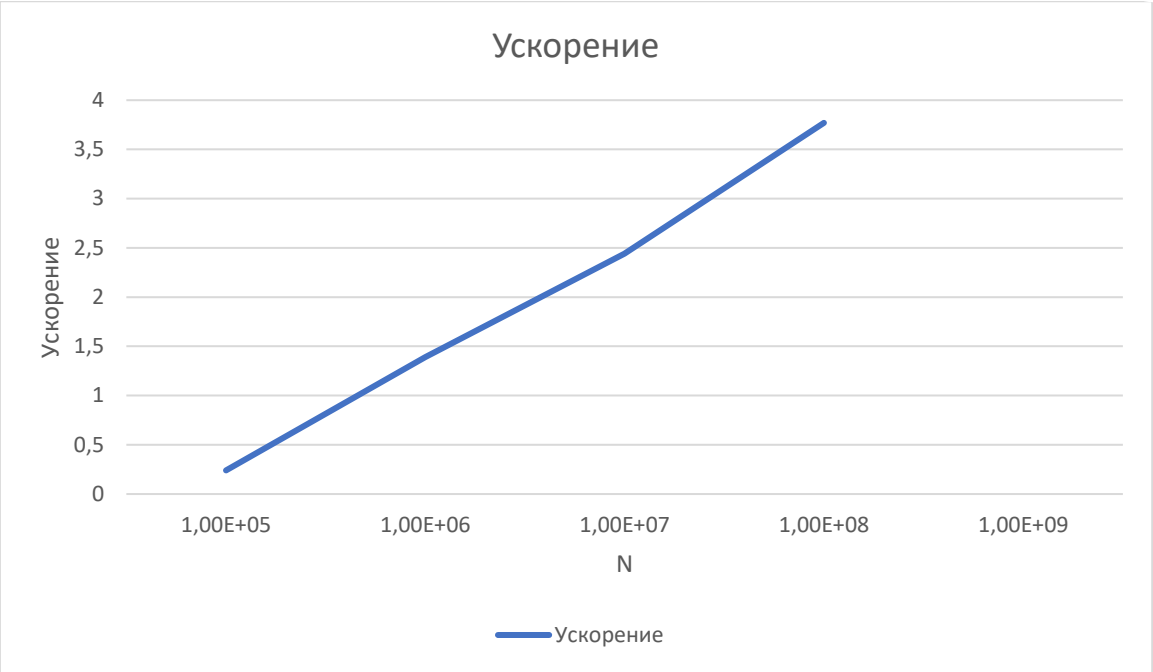


Видим, что гонки данных в нашем коде нет, т.к. о синхронизации заботится *reducer_opadd*.

Исследуем полученное ускорение

Сравним время выполнения последовательной и параллельной и версии.

N, количество точек разбиения	Время работы последовательной версии	Время работы параллельной версии	Ускорение
1e5	0,0001	0,0005	0,24
1e6	0,0014	0,0010	1,39
1e7	0,01134	0,1563	2,44
1e8	0,1145	0,0303	3,77
1e9	1,1601	0,3483	3,33



Как видим, при больших N мы можем достигнуть ускорения почти в 4 раза, что является пределом для данной машины, у которой 4 вычислительных ядра.