МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**

ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ, ФОРМУЛА СИМПСОНА, ВЫЧИСЛЕНИЕ ОПРЕДЕЛЕННЫХ И КРАТНЫХ ИНТЕГРАЛОВ

ОТЧЕТ О ПРАКТИКЕ

Студента 3 курса 311 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Аношкина Андрея Алексеевича

Проверил
Старший преподаватель ______________ М. С. Портенко

Саратов 2024

СОДЕРЖАНИЕ

1 Work 01

Задание

Реализуйте параллельные алгоритмы, использующие метод прямоугольников и формулу Симпсона для подсчета интегралов. Точные значения интегралов указаны для проверки численных вычислений. В случае, если в верхнем пределе интегрирования указан знак бесконечности, то в расчете необходимо заменить его на 10^6. Сравните время численного интегрирования для последовательной и параллельной реализации. Какое ускорение выполнения программы предоставляет переход к многопоточной версии?

Вариант задания 2:

$$\int_0^\infty \frac{dx}{1+x^2} = \frac{\pi}{2}$$

Метод прямоугольников, формула Симпсона

Метод прямоугольников геометрически заключается в том, что интеграл приближенно представляется в виде суммы площадей элементарных прямоугольников.

Для случая деления отрезка интегрирования на равные части и вычисления функции в центре отрезков:

$$\begin{cases} J = \int_a^b f(x)\,dx \approx \sum_{i=0}^{N-1}(h \cdot f(x_i)) = h \cdot \sum_{i=0}^{N-1} f(x_i) \\ x_i = a + i \cdot h + \frac{h}{2} \end{cases} \tag{1}$$

где N — количество отрезков интегрирования, а $h = (b-a)/N$

Ступенчатая (stair-case) аппроксимация гладких изогнутых поверхностей, возникает в результате дискретизации модели прямоугольной сеткой. Для борьбы со ступенчатой аппроксимацией может использоваться, например, квадратурная формула Симпсона:

$$\int_a^b f(x)\,dx \approx \frac{h}{3}[\,f(a) + f(b) + 4\sum_{k=1}^{N} f(a+(2k-1)h) + 2\sum_{k=1}^{N-1} f(a+2kh)] \tag{2}$$

где $h = \frac{b-a}{2N}$, $N \gg 1$.

Формула Симпсона геометрически заключается в том, что через три ординаты, отвечающие трем последовательным узлам сетки, проводится парабола и затем складываются получившиеся при этом площади элементарных криволинейных трапеций.

Реализация

Код решения приведен ниже:

```
#include <iostream>
#include <omp.h>
#include <time.h>
#define PI 3.14159265358979323846264338327950

using namespace std;

double f1(double x) {
	return 1.0 / (1 + x * x);
}

void integral_posl(const double a, const double b, const double h, double* res) {
	double sum = 0;
	int n = (int)((b - a) / h);

	for (int i = 0; i < n; ++i) {
		double x = a + i * h + h / 2;
		sum += f1(x) * h;
	}

	*res = sum;
}

void integral_paral(const double a, const double b, const double h, double* res) {
	double sum = 0;
	int n = (int)((b - a) / h);

	#pragma omp parallel for reduction(+: sum)
	for (int i = 0; i < n; ++i) {
		double x = a + i * h + h / 2;
		sum += f1(x) * h;
	}

	*res = sum;
}

void integral_Simpson(const double a, const double b, const double h, double* res) {
	double sum = f1(a) + f1(b);
	int n = (int)((b - a) / 2 * h);

	#pragma omp parallel for reduction(+: sum)
	for (int i = 1; i <= n; ++i) {
		double x = a + h * (2 * i - 1);
```

```
		sum += f1(x) * 4;
	}

	#pragma omp parallel for reduction(+: sum)
	for (int i = 1; i < n; ++i) {
		double x = a + 2 * i * h;
		sum += 2 * f1(x);
	}

	*res = h / 3 * sum;
}

double experiment(double* res, void f(const double a, const double b, const double h, double* res)) {
	double stime = 0, ftime = 0;
	double a = 0;
	double b = 1e+6;
	double h = 0.01;
	stime = clock();
	f(a, b, h, res);
	ftime = clock();

	return (ftime - stime) / CLOCKS_PER_SEC;
}

void calculate(void f(const double a, const double b, const double h, double* res)) {
	double avg_time = 0;
	double min_time = 0;
	double max_time = 0;
	double res = 0;
	int numbExp = 10;

	min_time = max_time = experiment(&res, f);
	avg_time = min_time / numbExp;
	for (int i = 0; i < numbExp - 1; ++i) {
		double time = experiment(&res, f);
		if (time > max_time)
			max_time = time;
		if (time < min_time)
			min_time = time;

		avg_time += time / numbExp;
	}

	cout << "Execution time: " << avg_time << "; " << min_time << "; " << max_time << "\n";
	cout.precision(8);
	cout << "Integral value: " << res << "\n";
}

int main() {

```

```
	bool calcPosl = true, calcParal = true, calcSimpson = true;

	// Последовательное вычисление

	if (calcPosl) {
		cout << "Posl:\n";
		calculate(integral_posl);
		cout << "\n";
	}

	// Параллельное вычисление

	if (calcParal) {
		cout << "Paral:\n";
		calculate(integral_paral);
		cout << "\n";
	}

	// Параллельное вычисление по формуле Симпсона

	if (calcSimpson) {
		cout << "Simpson:\n";
		calculate(integral_Simpson);
		cout << "\n";
	}

	system("pause");
	return 0;
}
```

Результаты работы

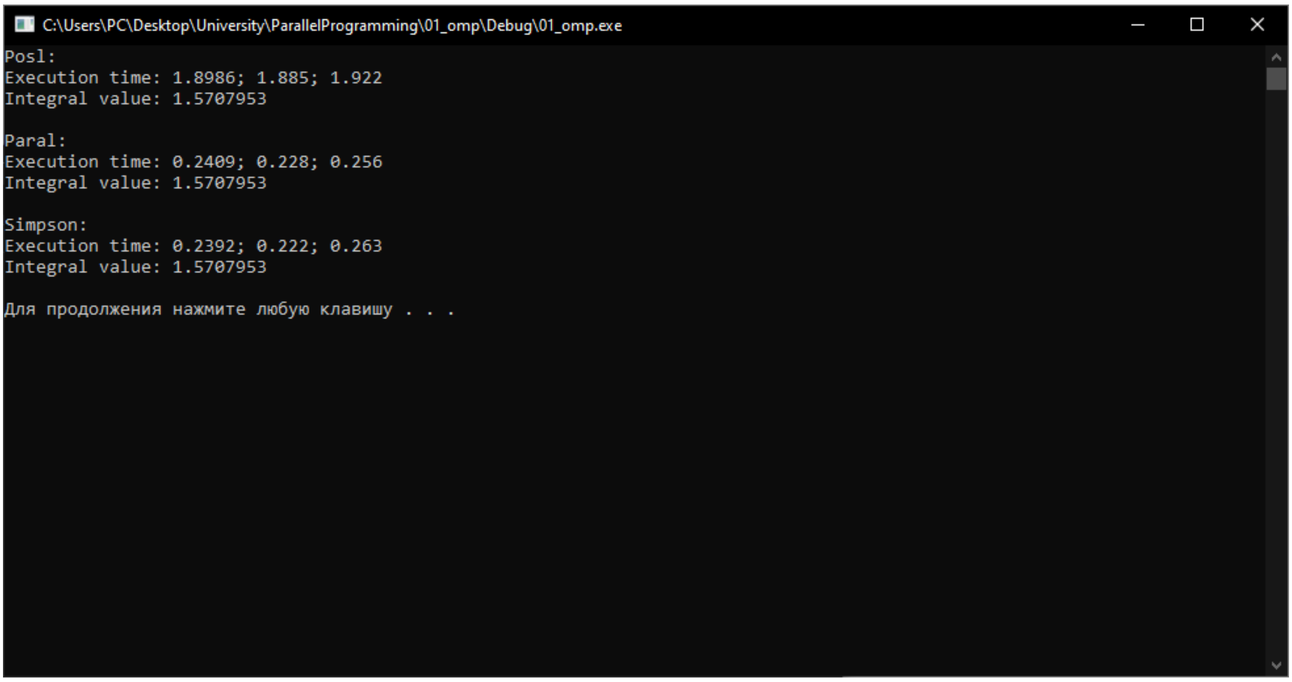

Рисунок 1 – Work-1

Значение ускорения выполнения программы при переходе к многопоточной версии:

$$a = \frac{timeSeq}{timePar} = \frac{1.8986}{0.2409} \approx 7.8812$$

Значение ускорения выполнения программы при переходе к многопоточной версии по формуле Симпсона:

$$a = \frac{timeSeq}{timePar} = \frac{1.8986}{0.2392} \approx 7.9373$$

Характеристики устройства

Процессор: Intel(R) Core(TM) i5-10400F

Ядер: 6

Оперативная память: 16 Гб