Параллельный подсчет интеграла с помощью OpenMP

Нужно подсчитать значение интеграла $\int_{0.01}^{1} cos(\frac{1}{x^2})dx$

Алгоритм выполнения:

1. Вычисляем интеграл на всем отрезке $[a, b]$ по формуле трапеции
   $I_{[a,b]} = \frac{(f(a)+f(b))*(b-a)}{2}$
2. Разбиваем интервал на 2 равные части и вычисляем интеграл по формуле трапеции на каждой части: $I_{[a,m]}, I_{[m,b]}$
3. Сравниваем разность с погрешностью EPS: $|(I_{[a,m]} + I_{[m,b]}) - I_{[a,b]}|$
4. Если $|(I_{[a,m]} + I_{[m,b]}) - I_{[a,b]}| < EPS$, то сумма интегралов и есть значение интеграла
5. Иначе, если $|(I_{[a,m]} + I_{[m,b]}) - I_{[a,b]}| > EPS$, то на каждой части проводим заново те же действия, но с ошибкой уже $\frac{EPS}{2}$ (получается рекурсия). Я решила брать такую ошибку, потому что потом при сложении ошибки складываются, а суммарная не должна превышать EPS

В своей программе я реализовала подход такой: после деления отрезка пополам, вычисление интеграла на каждом отрезке считается задачей и помещается в пул задач, откуда каждый свободный tread может брать задачу и выполнять ее. Если задач в пуле всегда больше, чем tread'ов (в несколько раз), то будет выполняться балансировка

Такой подход для решения я выбрала, потому что выполняется подбор нужного шага на каждом участке. Если у нас пологий участок, то и шаг будет больше, а если участок с большими производными, то и шаг будет маленьким

Результаты:

Аналитическое решение: -0.092475

Для num_threads(4):

```
(base) alina@MacBook-Pro-Alina Парпрог % gcc -Xpreprocessor -fopenmp -lomp -o Integral Integral
.c
(base) alina@MacBook-Pro-Alina Парпрог % ./Integral
accurate result: -0.092475
approximate result: -0.0924757
Time: 0.233048
eps: 0.0000100
```

```
(base) alina@MacBook-Pro-Alina Парпрог % gcc -Xpreprocessor -fopenmp -lomp -o Integral Integral
.c
(base) alina@MacBook-Pro-Alina Парпрог % ./Integral
accurate result: -0.092475
approximate result: -0.0924754
Time: 0.689790
eps: 0.0000010
(base) alina@MacBook-Pro-Alina Парпрог %
```

```
(base) alina@MacBook-Pro-Alina Парпрог % gcc -Xpreprocessor -fopenmp -lomp -o Integral Integral
.c
(base) alina@MacBook-Pro-Alina Парпрог % ./Integral
accurate result: -0.092475
approximate result: -0.0924754
Time: 2.199532
eps: 0.0000001
(base) alina@MacBook-Pro-Alina Парпрог %
```

Без OpenMP:

```
(base) alina@MacBook-Pro-Alina Парпрог % ./Integral_no_omp
accurate result: -0.092475
approximate result: -0.0924757
Time: 0.491244
eps: 0.0000100
(base) alina@MacBook-Pro-Alina Парпрог %
```

```
(base) alina@MacBook-Pro-Alina Парпрог % ./Integral_no_omp
accurate result: -0.092475
approximate result: -0.0924754
Time: 1.486775
eps: 0.0000010
(base) alina@MacBook-Pro-Alina Парпрог %
```

```
(base) alina@MacBook-Pro-Alina Парпрог % ./Integral_no_omp
accurate result: -0.092475
approximate result: -0.0924754
Time: 4.625754
eps: 0.0000001
(base) alina@MacBook-Pro-Alina Парпрог %
```

Анализ ускорения и эффективности

Ошибка	T_1	T_4	$S_4 = \frac{T_1}{T_4}$	$E_4 = \frac{S_4}{4}$
0.0000100	0,49	0,23	2,1304347826087	0,532608695652175
0.0000010	1,4	0,68	2,05882352941176	0,51470588235294
0.0000001	4,6	2,18	2,11009174311927	0,527522935779818