**НИЯУ “МИФИ”**

**Индивидуальное задание №8**

Группа: М19-117

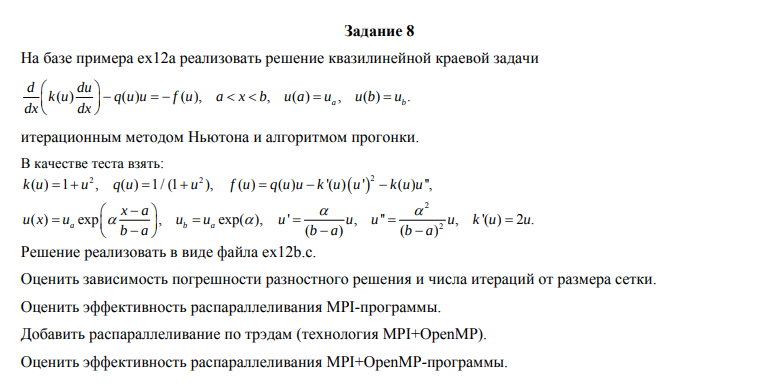
Студенты: Кайгородов Александр

Сергеев Олег

Преподаватель: Поляков С.В.

Москва, 2020

***Задание 8:***



***Решение:***

*1. Решение прикреплено в сообщении в виде двух файлов. ex12b.c и ex12b\_th.c*

*2. Оценка зависимости погрешности решения от размера сетки*

|  |  |
| --- | --- |
| *Размер сетки, ед. узлов* | *Погрешность расчета, dmax* |
| 10 | dmax=2.830711e-03 |
| 100 | dmax=2.869906e-05 |
| 500 | dmax=1.148111e-06 |
| 1000 | dmax=2.870285e-07 |
| 5000 | dmax=1.145606e-08 |
| 10\_000 | dmax=2.862016e-09 |
| 50\_000 | dmax=5.845533e-10 |
| 100\_000 | dmax=1.776295e-09 |

Как видно из таблицы значений погрешности полученных при работе программы ex11c.c при одном работающем процессе, оптимальным значением разбиения сетки является величина 50\_000 узлов. Расчетная схема наращивает ошибку вычислений при величине разбиения от 50k+.

*3. Оценка эффективности распараллеливания MPI-программы.*

В алгоритме используется метод прогонки теоретическая эффективность которого составляет от 33 до 100%. Оценить эффективность можно следующим образом.

Где p - число запущенных процессов, tp - время решения исходной задачи по параллельному алгоритму на p процессорах, t1 - время решения исходной задачи на одном процессоре.



Проведем натурный эксперимент на конкретной задаче:

t1 = 1.6343 c на сетке 1000 узлов.

|  |  |  |
| --- | --- | --- |
| Число процессов | Время на выполнение, c | Эффективность |
| 2 | 0.8817 | 0.927 |
| 4 | 0.4924 | 0.828 |
| 7 | 0.3420 | 0.683 |
| 10 | 0.2381 | 0.686 |
| 20 | 0.1790 | 0.456 |
| 40 | 0.1719 | 0.238 |

Как и ожидалось, время выполнения программы уменьшается, эффективность падает, но все в разумных пределах. В момент когда колличество процессов превышает ~20 начинается возрастающий рост времени на обмен сообщениями между процессами и как следствие резко падает эффективность распараллеливания.

*4. Добавить распараллеливание по трэдам (технология MPI+OpenMP).*

Файл прикреплен в сообщении.

*5. Оценить эффективность распараллеливания MPI+OpenMP-программы.*

Идея распараллеливания по тредам заключалась в том, что-бы выбрать отдельную часть программы с использованием метода прогонки, выделить эту часть в отдельную функцию, определить ее выполнение на каждом отдельном процессе, но уже с применением распараллеливания по тредам. Главный поток (работающий с первым узлом процесса) занимается передачей информации остальным процессам, что собственно раньше выполнялось процессом самостоятельно.

Использование многопоточности приводит к увеличению используемой памяти при вычислении на процессоре вследствие введения дополнительных переменных и контейнеров, но чисто вычислений остается прежним. Как итог, для оценки сложности данного алгоритма с использованием многопоточности сокращает число вычислений на n-тредов раз, но время работы все же слегка ограничено периодическим ожиданием отработки первого потока процесса остальными, перед сведением результатов в единый результат и временем работы этого потока на задачу выдачи или сбора результатов от остальных процессов.

Полученные результаты времени исполнения программы:

|  |  |  |  |
| --- | --- | --- | --- |
| P / T | t1 | t2 | t3 |
| 1/1 |  |  |  |
| 1/2 |  |  |  |
| 1/4 |  |  |  |
| 2/2 |  |  |  |
| 2/4 |  |  |  |
| 2/6 |  |  |  |

По полученным результатам сложно говорить об ускорении четко. Видно увеличение времени исполнения программы при распараллеливании на треды при одном процессе, но с другой стороны получаем уже относительно заметное ускорение при распараллеливании на треды при выполнении на двух процессах. При выполнении на двух процессах, общее число выполняемых нитей равняется np\*nt.