Московский Государственный Университет им. М.В. Ломоносова
Факультет Вычислительной Математики и Кибернетики
Кафедра Суперкомпьютеров и Квантовой Информатики

Отчёт.

Решение СЛАУ методом BiCGSTAB

Работу выполнил
Пилюгин В.И., 523 группа

15 октября 2018 г.

Постановка задачи и формат данных

Задача: Многопоточная реализация солвера BiCGSTAB для СЛАУ с разреженной матрицей, заданной в формате CSR. Необходимо реализовать алгоритм генерации матрицы в формате CSR, в качестве расчетной области использовать трехмерную декартову решетку.
Реализация: Были реализованы несколько вспомогательных функций:
double dot(Vector X, Vector Y) - скалярное произведение двух векторов
void axpby(Vector X, Vector Y, double a, double b) - операция X = a*X+b*Y
void SpMV(CSRMatrix A, Vector X, Vector Y) - операция Y = A*X
void fill_with_constant(Vector X, const int c) - операция X[i] = c for i = 0..len(X)-1
void assign_vector(Vector X, Vector Y) - операция X[i] = Y[i] for i = 0..len(X)-1
а также главный алгоритм решения СЛАУ, который использует все эти базовые операции:

int bicgstab_solve(CSRMatrix A, Vector BB, Vector XX, double tol, int maxit, int info) - решение СЛАУ с матрицей A, правой частью BB, вектором решения XX, tol - отношение нормы невязки к норме правой части, maxit - максимальное число итераций, info - флаг отладки

Данные организованы следующим образом:

```
typedef struct CSRMatrix {
  int *IA; // содержит позицию начала данных i-й строки в массивах A и JA
  int *JA; // хранит номера столбцов ненулевых коэф. подряд по всем строкам
  double *A; // хранит значения ненулевых коэффициентов подряд по всем строкам
  int N; // число строк в матрице
} CSRMatrix;

typedef struct Vector {
  double *data; // хранит значения элементов
  int size; // размер вектора
} Vector;
```

Компиляция и запуск

Формат командной строки: *main [NX] [NY] [NZ] [TOL] [MAXIT] [THREADS] [QA]*
NX, NY, NZ - размер решетки топологии для генерации матрицы размера *NxN, N=nx*ny*nz*
TOL - невязка относительно нормы правой части
MAXIT - максимальное число итераций
THREADS - число нитей
QA - флаг тестирования базовых операций
Пример:
mpisubmit.bg -n 1 main 100 100 100 1e-6 50 1 0

На ноутбуке: компиляция с помощью *gcc*, с ключом *-O3*. Далее полученный файл запускается командой *./main <ARGS>*.
На BlueGene: использовался компилятор *bgxlc_r,* с ключом *-O5*. Полученный исполняемый файл ставился в очередь на запуск скриптом *mpisubmit.bg* в режиме *SMP*.
На Polus: использовался компилятор *xlc_r,* с ключами *-O5 -qarch=pwr8 -qhot.* Полученный исполняемый файл ставился в очередь на запуск скриптом *mpisubmit.pl*.

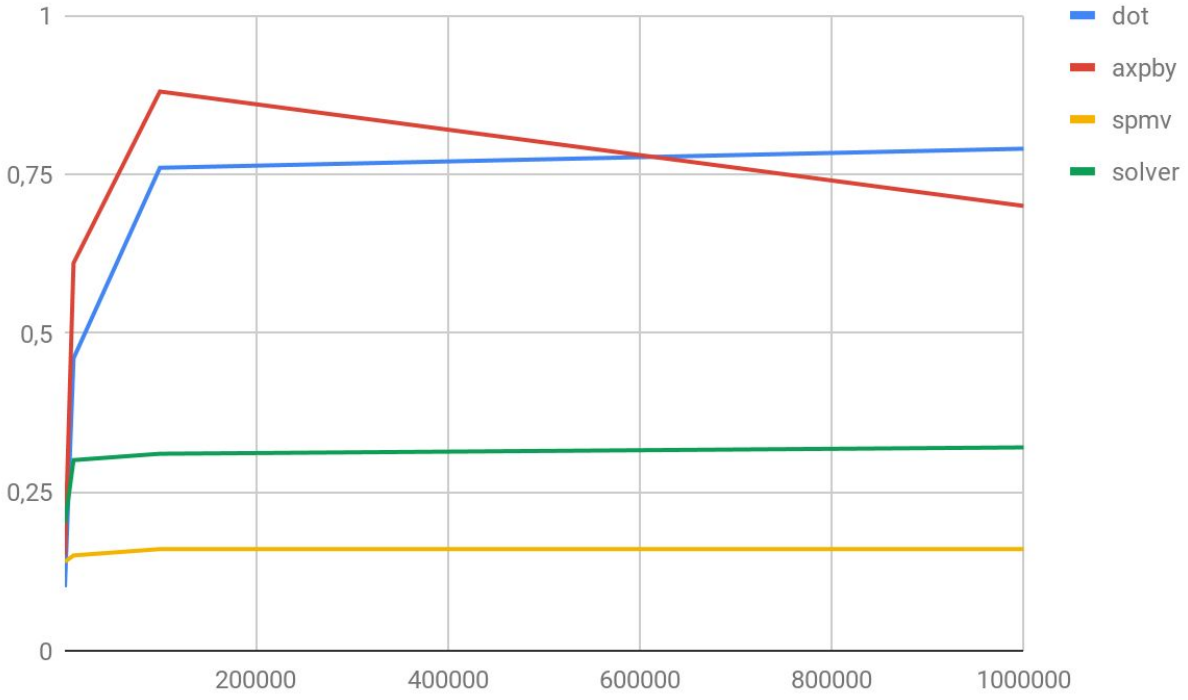
Исследование производительности

IBM Polus: - параллельная вычислительная система, состоящая из 5 вычислительных узлов. Каждый узел содержит два 10-ядерных процессора IBM POWER8 (каждое ядро имеет 8 потоков), всего 160 потоков.
Производительность: производительность кластера (Tflop/s): 55,84 (пиковая), 40,39 (Linpack). IBM Power8 CPU: 329.28 GFlop/sec.

IBM BlueGene/P: - массивно-параллельная вычислительная система, которая состоит из двух стоек, включающих 8192 процессорных ядер с пиковой производительностью 27,9 терафлопс.
Производительность: пиковая производительность одного узла: 4 cores x 3,4 GFlop/sec per core = 13,6 GFlop/sec.

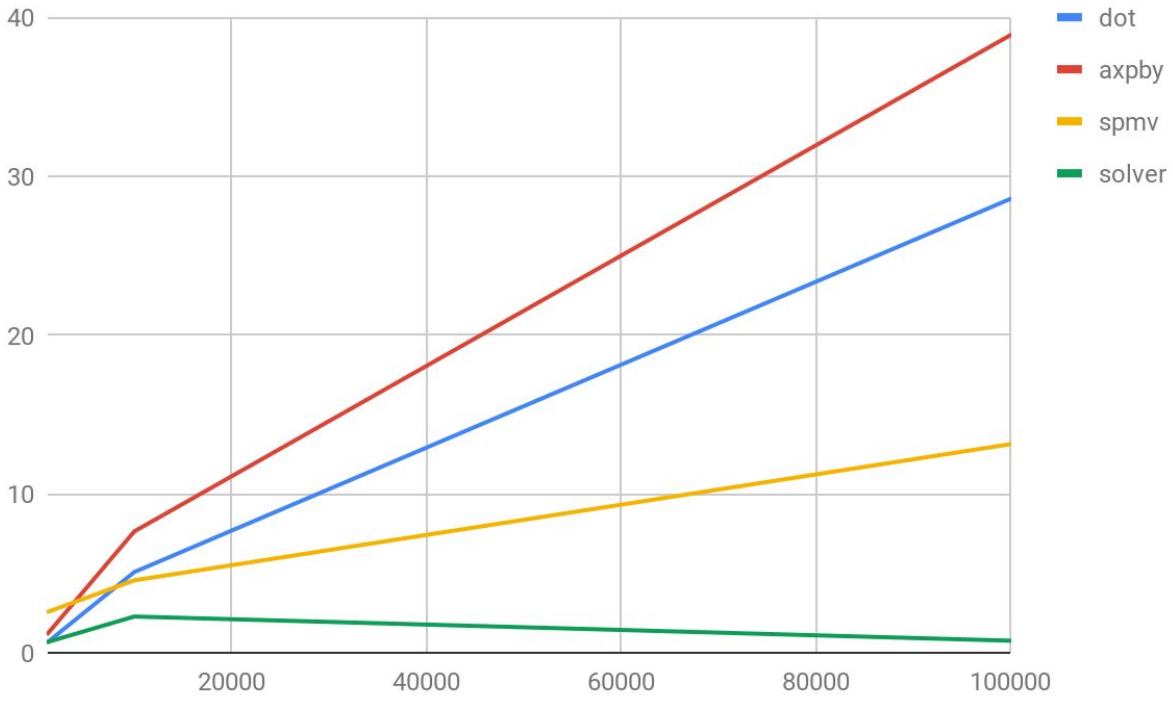
Результаты выполнения операций на **BlueGene** при однопоточном выполнении (GFLOPS):

N	1000	10000	100000	1000000
dot	0,1	0,46	0,76	0,79
axpby	0,15	0,61	0,88	0,7
spmv	0,14	0,15	0,16	0,16
solver	0,2	0,3	0,31	0,32



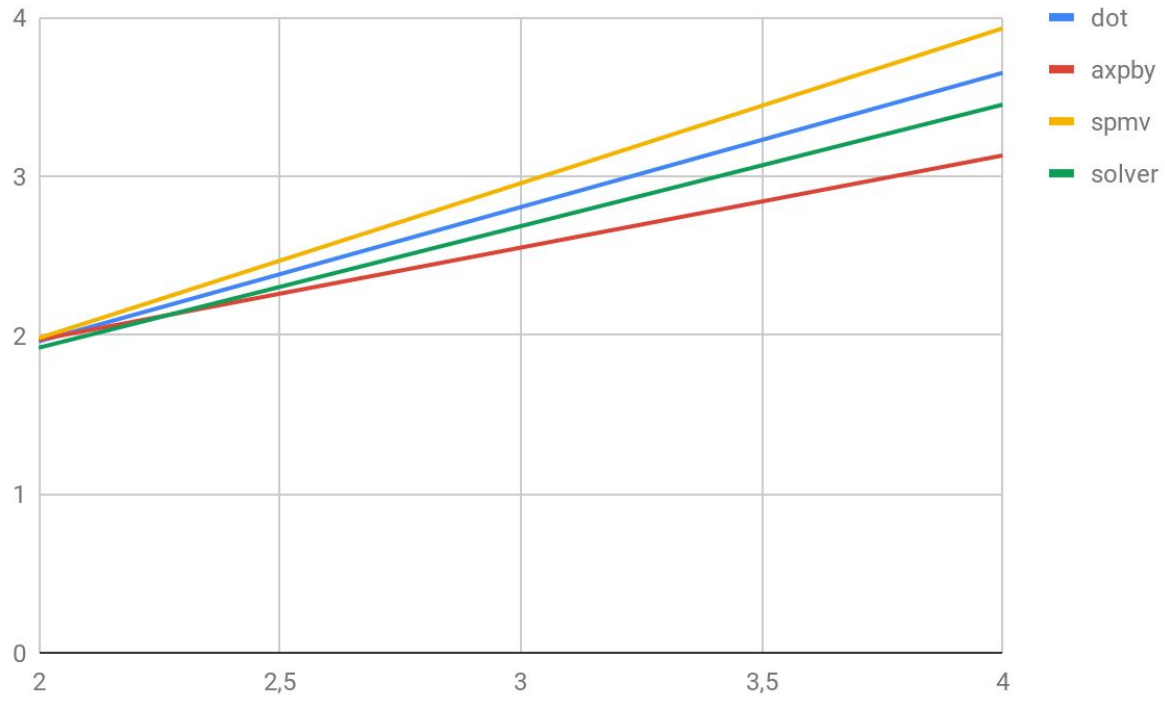
Результаты выполнения операций на **Polus** при однопоточном выполнении (GFLOPS):

N	1000	10000	100000	1000000
dot	0,62	5,08	28,57	3,23
axpby	1,13	7,64	38,88	3,4
spmv	2,55	4,56	13,13	11,71
solver	0,66	2,28	0,76	3,43



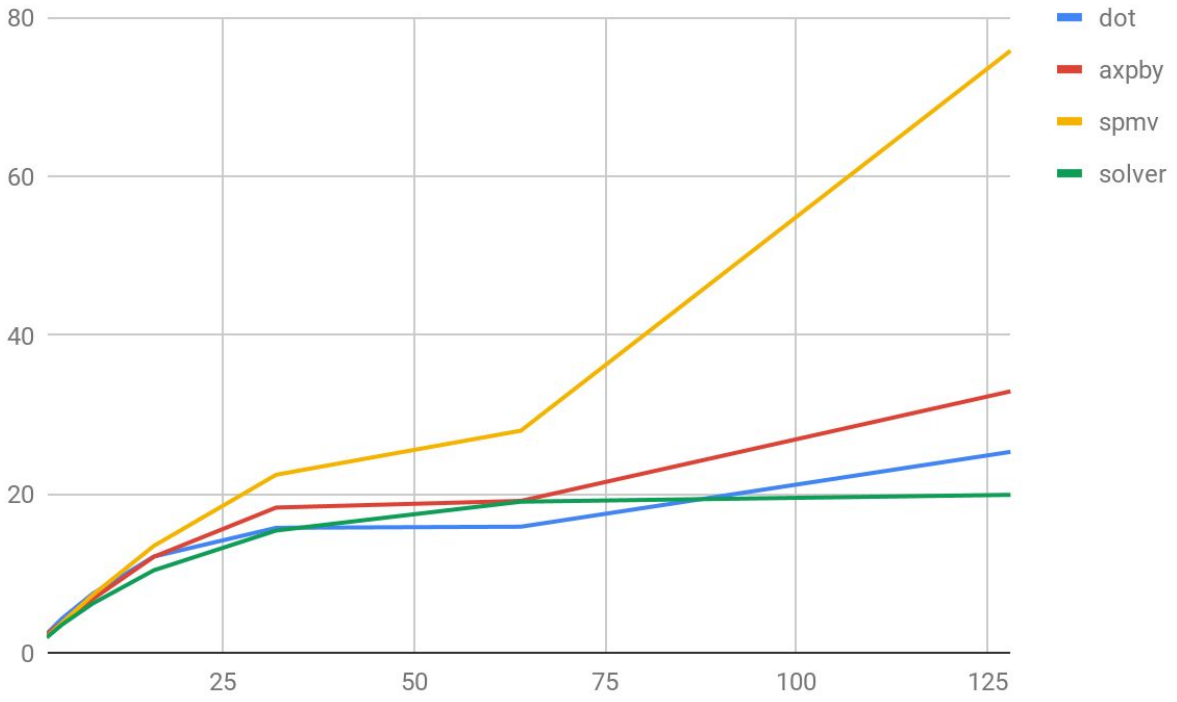
Результаты выполнения операций на **BlueGene** при многопоточном выполнении (ускорение) при Nx = Ny = Nz = 100:

Threads	2	4
dot	1,96	3,65
axpby	1,97	3,13
spmv	1,98	3,93
solver	1,92	3,45



Результаты выполнения операций на **Polus** при многопоточном выполнении (ускорение) при Nx = Ny = Nz = 100:

Threads	2	4	8	16	32	64	128
dot	2,38	4,31	7,42	12,11	15,73	15,87	25,29
axpby	2,32	3,8	6,75	12,08	18,28	19,12	32,91
spmv	1,97	3,79	7,26	13,45	22,41	27,96	75,79
solver	1,92	3,54	6,19	10,38	15,39	19,02	19,88



Анализ полученных результатов

Максимальная достигаемая производительность:

BlueGene:
dot: 1.48 Gflops, *11% от пиковой*
axpby: 1.64 Gflops, *12% от пиковой*
spmv: 0.31 Gflops, *2% от пиковой*
solver: 0.60 Gflops, *4% от пиковой*

Polus:
dot: 37.77 Gflops, *11% от пиковой*
axpby: 71.78 Gflops, *22% от пиковой*
spmv: 14.84 Gflops, *5% от пиковой*
solver: 6.61 Gflops, *2% от пиковой*