МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПРИ СОДРУЖЕСТВЕ НГТУ И НГУ

НГТУ НЭТИ

Новосибирский государственный университет

*НАСТОЯЩАЯ НАУКА

Кафедра Параллельных Вычислительных Технологий

Анализ производительности параллельных программ при помощи специальных инструментов

Студент: Саяпин Матвей Павлович

Руководитель проекта: Власенко Андрей Юрьевич

Новосибирск 2020

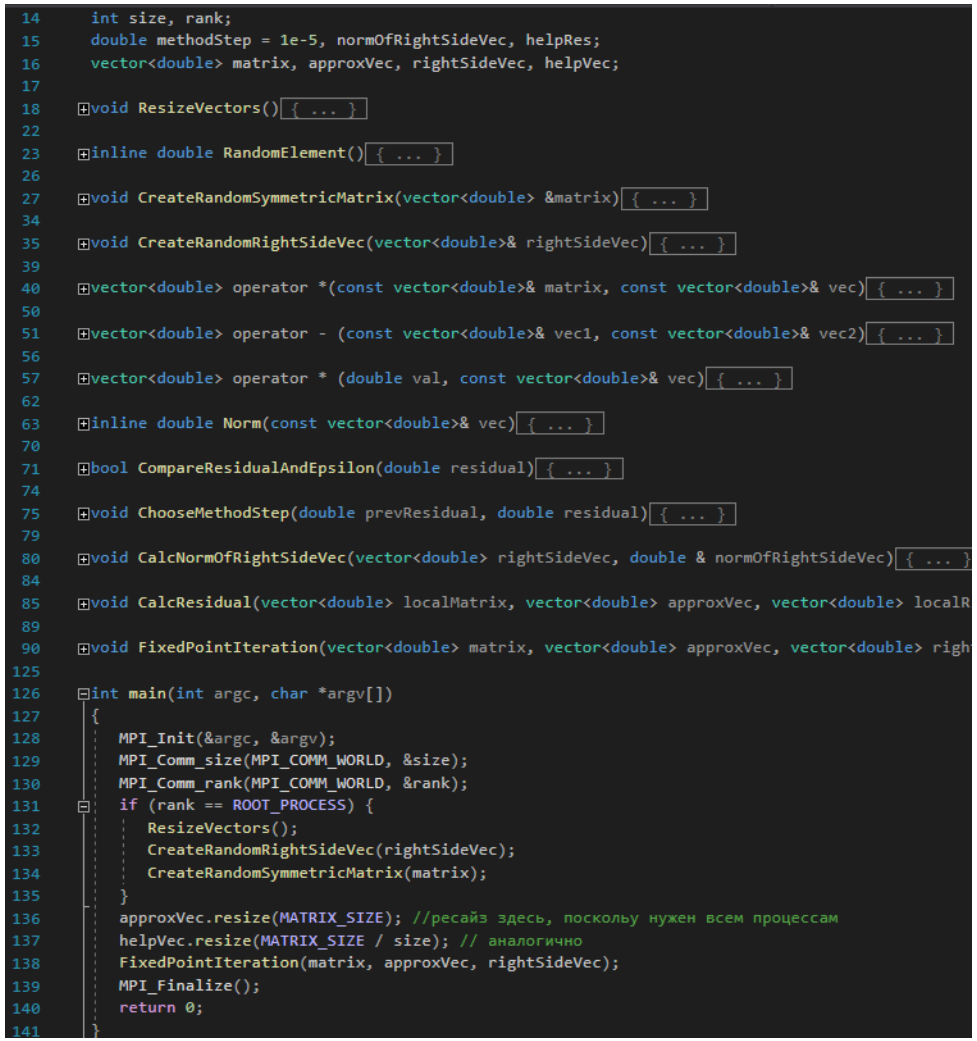
Цели и задачи

- Изучение профессиональных инструментов трассировки и профилирования параллельных программ (*TAU - Tuning and Analysis Utilities*). Установка и тестирование инструментальных средств на учебных параллельных программах, использующих *MPI*. Применение полученных навыков к анализу системы фрагментированного программирования LuNA.

Этапы выполнения проекта

- Разработка MPI-программы решения СЛАУ методом простой итерации
- Установка и настройка средств TAU - Tuning and Analysis Utilities на кафедральный сервер и кластер НГУ
- Разработка эквивалентной LuNA-программы
- Первичный анализ узких мест в системе LuNA при помощи указанных средств на примере созданной программы

MPI-программа



```
14    int size, rank;
15    double methodStep = 1e-5, normOfRightSideVec, helpRes;
16    vector<double> matrix, approxVec, rightSideVec, helpVec;
17
18  ⊞void ResizeVectors() { ... }
22
23  ⊞inline double RandomElement() { ... }
26
27  ⊞void CreateRandomSymmetricMatrix(vector<double> &matrix) { ... }
34
35  ⊞void CreateRandomRightSideVec(vector<double>& rightSideVec) { ... }
39
40  ⊞vector<double> operator *(const vector<double>& matrix, const vector<double>& vec) { ... }
50
51  ⊞vector<double> operator - (const vector<double>& vec1, const vector<double>& vec2) { ... }
56
57  ⊞vector<double> operator * (double val, const vector<double>& vec) { ... }
62
63  ⊞inline double Norm(const vector<double>& vec) { ... }
70
71  ⊞bool CompareResidualAndEpsilon(double residual) { ... }
74
75  ⊞void ChooseMethodStep(double prevResidual, double residual) { ... }
79
80  ⊞void CalcNormOfRightSideVec(vector<double> rightSideVec, double & normOfRightSideVec) { ...
84
85  ⊞void CalcResidual(vector<double> localMatrix, vector<double> approxVec, vector<double> localR
89
90  ⊞void FixedPointIteration(vector<double> matrix, vector<double> approxVec, vector<double> righ
125
126 ⊟int main(int argc, char *argv[])
127   {
128     MPI_Init(&argc, &argv);
129     MPI_Comm_size(MPI_COMM_WORLD, &size);
130     MPI_Comm_rank(MPI_COMM_WORLD, &rank);
131   ⊟ if (rank == ROOT_PROCESS) {
132         ResizeVectors();
133         CreateRandomRightSideVec(rightSideVec);
134         CreateRandomSymmetricMatrix(matrix);
135     }
136     approxVec.resize(MATRIX_SIZE); //ресайз здесь, поскольу нужен всем процессам
137     helpVec.resize(MATRIX_SIZE / size); // аналогично
138     FixedPointIteration(matrix, approxVec, rightSideVec);
139     MPI_Finalize();
140     return 0;
141   }
```

- Разработана программа для решения СЛАУ методом простой итерации
- Метод очень “капризный”, скорость сходимости зависит от числа обусловленности и размерности матрицы, а также от числа обусловленности.
- Для распараллеливания программы были использованы коллективные операторы MPI
- Тесты проводились на размерности 5000
- Влиять на число обусловленности можно через усиление главной диагонали
- Сделаем ее такой, чтобы метод сходился примерно за 100 итераций

Средства для профилирования программы

- Была выбрана утилита TAU - Tuning and Analysis Utilities (2.30.1)
- Установка проводилась на кафедральном сервере и на кластере
- При установке в качестве компиляторов были выбраны

  на кафедральном: mpich 3.2, g++ 5.5.0

  на кластере: intel_composer_xe_2015.2.164, gcc4.8.5, intel_mpi-4.1.3.049
- Для вывода информации на экран был использован Xming
- В bashrc были помещены пути к скриптам, для профилировки программы

Профилирование MPI-программы

2 процесса

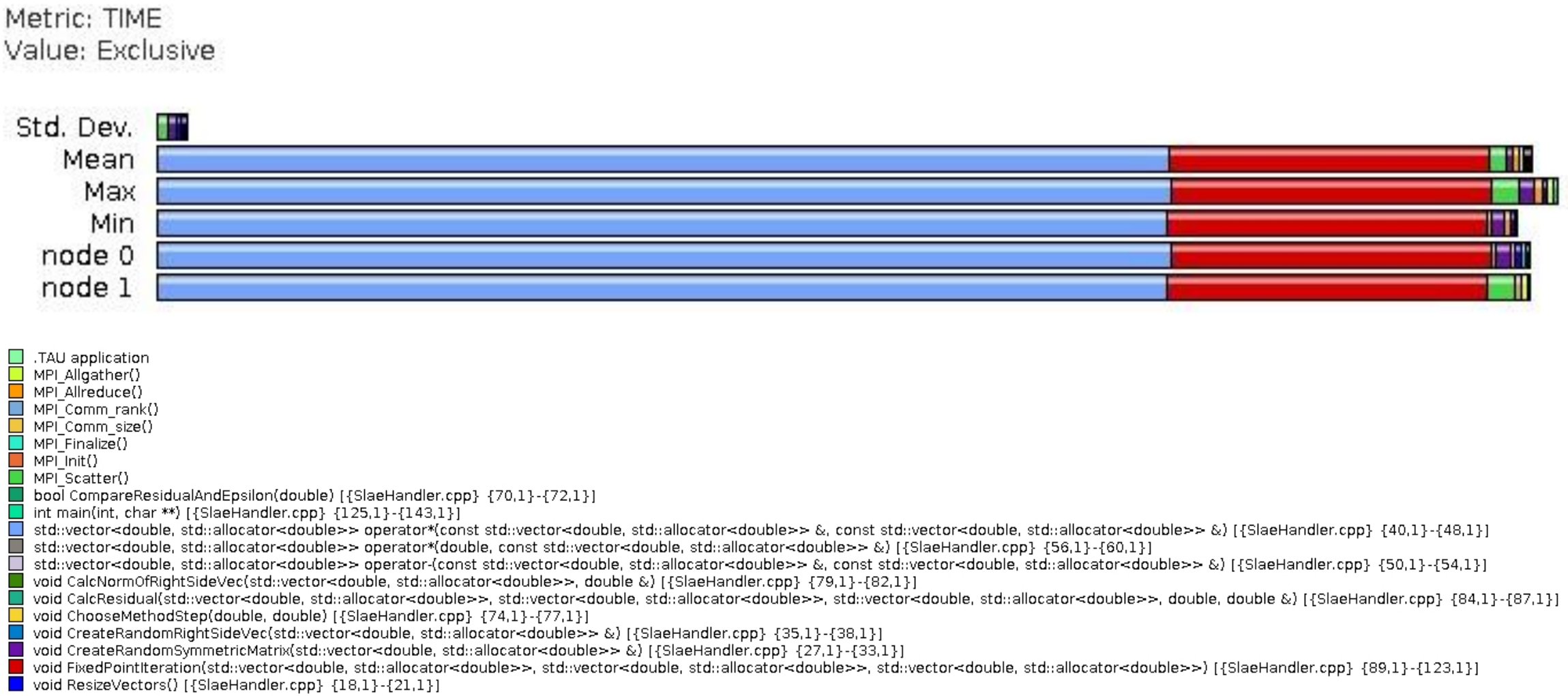


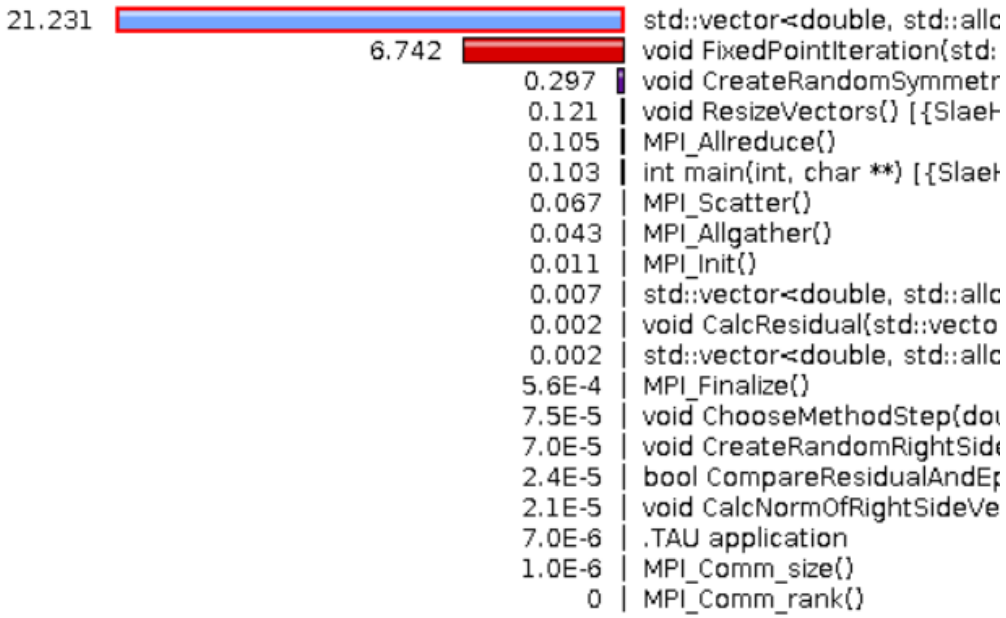
Таблица вызовов функций

Для двух процессов

Name △	Exclusive TIME	Inclusive TIME	Calls	Child Calls
.TAU application	0	28.733	1	1
MPI_Allgather()	0.105	0.105	95	0
MPI_Allreduce()	0.152	0.152	96	0
MPI_Comm_rank()	0	0	1	0
MPI_Comm_size()	0	0	1	0
MPI_Finalize()	0.007	0.007	1	0
MPI_Init()	0.011	0.011	1	0
MPI_Scatter()	0.579	0.579	2	0
bool CompareResidualAndEpsilon(double) [{SlaeHandler.cpp} {70,1}-{72,1}]	0	0	94	0
int main(int, char **) [{SlaeHandler.cpp} {125,1}-{143,1}]	0	28.733	1	5
std::vector<double, std::allocator<double>> operator*(const std::vector<double, std:	21.14	21.14	189	0
std::vector<double, std::allocator<double>> operator*(double, const std::vector<dou	0.002	0.002	94	0
std::vector<double, std::allocator<double>> operator-(const std::vector<double, std:	0.007	0.007	283	0
void CalcNormOfRightSideVec(std::vector<double, std::allocator<double>>, double &)	0	0	1	1
void CalcResidual(std::vector<double, std::allocator<double>>, std::vector<double, st	0.002	10.79	95	285
void ChooseMethodStep(double, double) [{SlaeHandler.cpp} {74,1}-{77,1}]	0	0	94	0
void FixedPointIteration(std::vector<double, std::allocator<double>>, std::vector<dou	6.728	28.715	1	757

График вызова функций

0 процесс



1 процесс

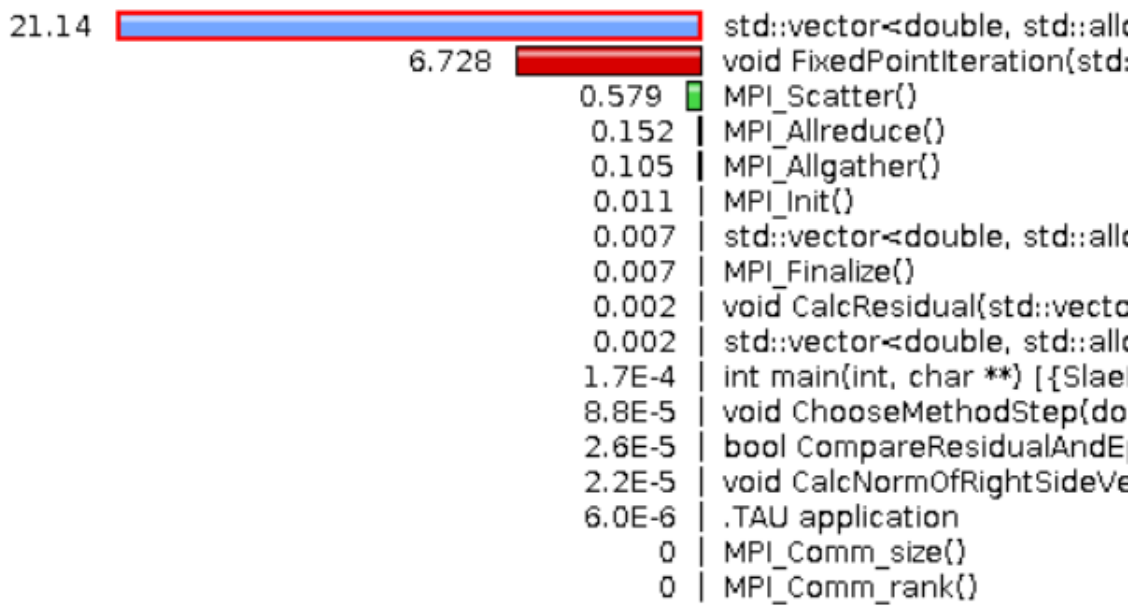
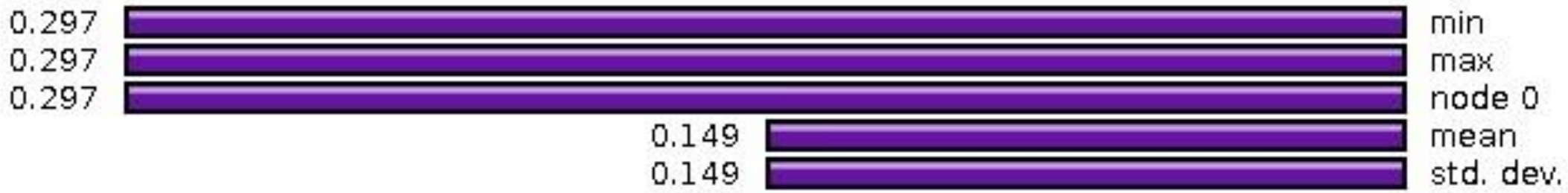


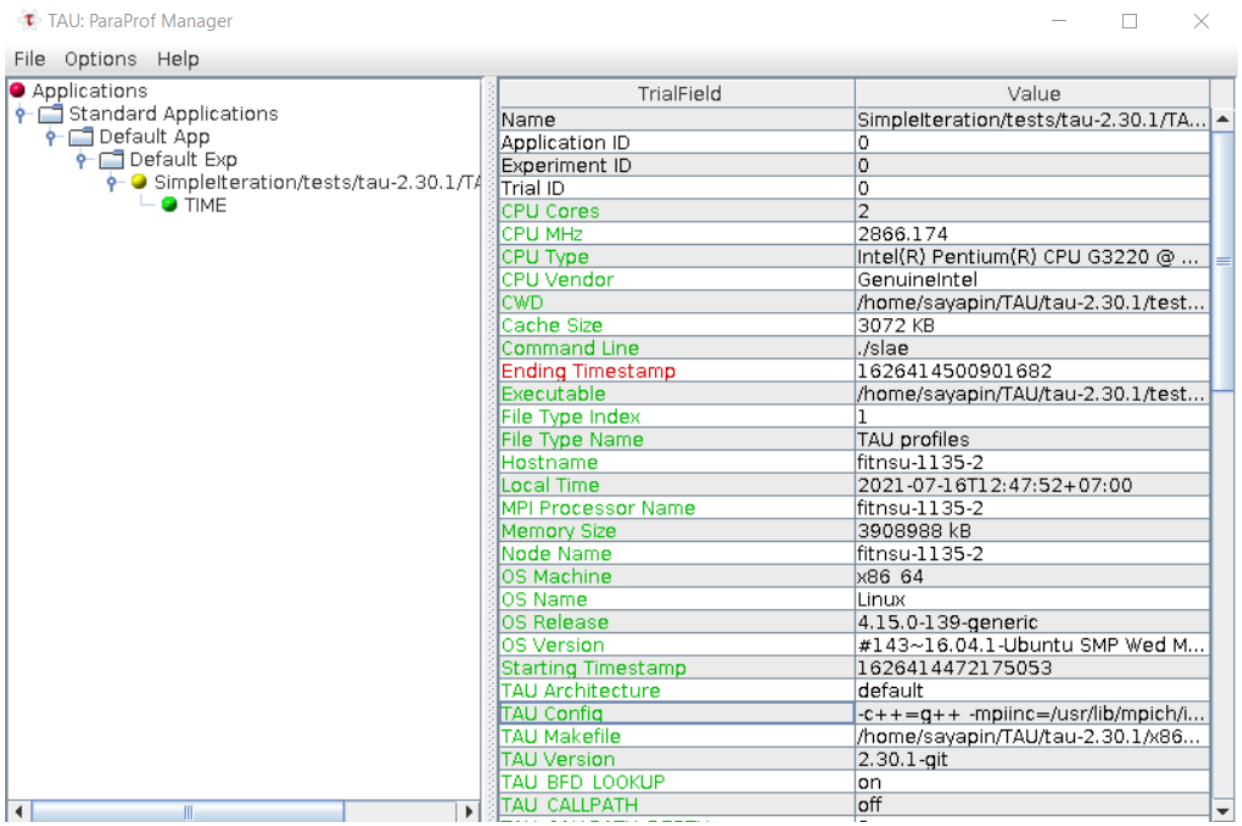
Таблица вызова функции

Name: void CreateRandomSymmetricMatrix(std::vector<double, std::allocator<double>> &)
[{SlaeHandler.cpp} {27,1}-{33,1}]
Metric Name: TIME
Value: Exclusive
Units: seconds

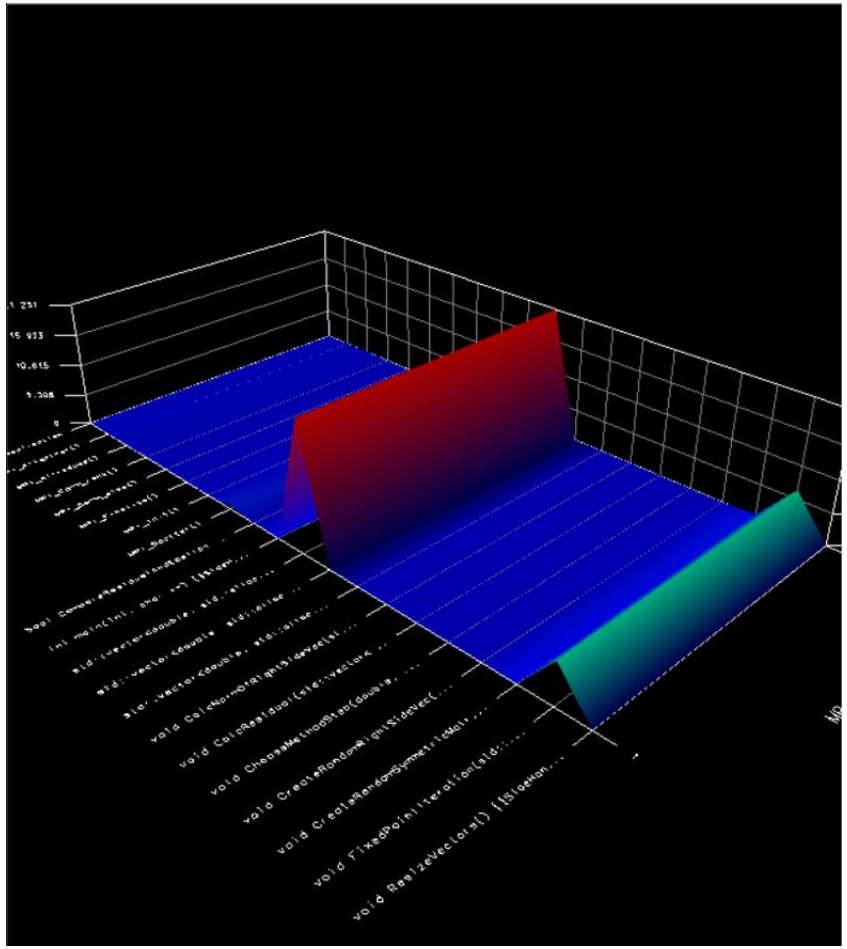


Функция создания матрицы выполняется только на ROOT(0) процессе. На графике видно, что второй процесс не выполняет эту функцию.

3D визуализация и сводная информация

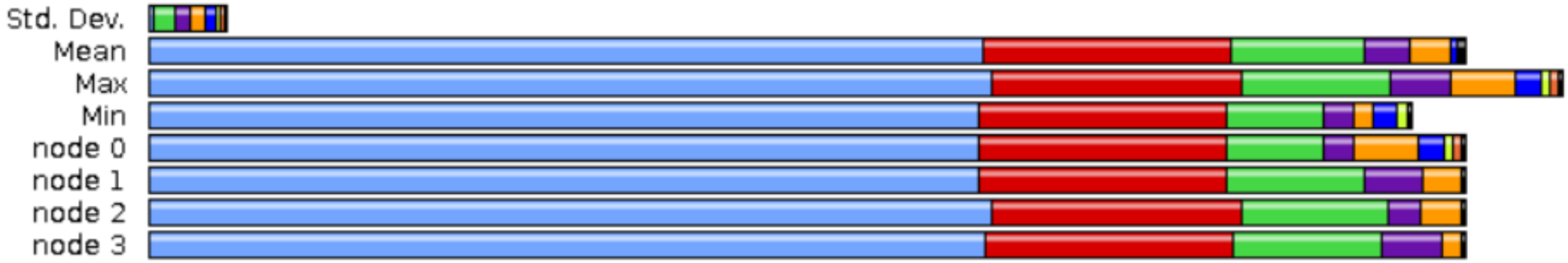


TrialField	Value
Name	SimpleIteration/tests/tau-2.30.1/TA...
Application ID	0
Experiment ID	0
Trial ID	0
CPU Cores	2
CPU MHz	2866.174
CPU Type	Intel(R) Pentium(R) CPU G3220 @ ...
CPU Vendor	GenuineIntel
CWD	/home/sayapin/TAU/tau-2.30.1/test...
Cache Size	3072 KB
Command Line	./slae
Ending Timestamp	1626414500901682
Executable	/home/sayapin/TAU/tau-2.30.1/test...
File Type Index	1
File Type Name	TAU profiles
Hostname	fitnsu-1135-2
Local Time	2021-07-16T12:47:52+07:00
MPI Processor Name	fitnsu-1135-2
Memory Size	3908988 kB
Node Name	fitnsu-1135-2
OS Machine	x86_64
OS Name	Linux
OS Release	4.15.0-139-generic
OS Version	#143~16.04.1-Ubuntu SMP Wed M...
Starting Timestamp	1626414472175053
TAU Architecture	default
TAU Config	-c++=g++ -mpiinc=/usr/lib/mpich/i...
TAU Makefile	/home/sayapin/TAU/tau-2.30.1/x86...
TAU Version	2.30.1-git
TAU_BFD_LOOKUP	on
TAU_CALLPATH	off



Профилирование MPI-программы

4 процесса



LuNA-программа

```
sub main()
{
    df matrix, localMatrix, rightSideVec, localRightSideVec, approxVec, localApproxVec, normOfRightSideVec,

    Init(4, size, normOfRightSideVec[0], approxVec[0], prevResidual[0], residual[0]);

    CreateRandomSymmetricMatrix(matrix);
    CreateRandomRightSideVec(rightSideVec);

    for split = 0..size-1 {
        SplitMatrix(split, matrix, localMatrix[split]);
        SplitRightSideVec(split, rightSideVec, localRightSideVec[split]);
        CreateLocalApproxVec(localApproxVec[split]);
    }


    for split = 0..size-1 {
        Norm(localRightSideVec[split], localNormOfRightSideVec[split]);
        Sum(normOfRightSideVec[split],localNormOfRightSideVec[split], normOfRightSideVec[split + 1]);
        AssemblyApproxVec(split, approxVec[split], localApproxVec[split], approxVec[split + 1]);
    }

    for split = 0..size-1 {
        CalcResidual(localMatrix[split], approxVec[size], localRightSideVec[split], normOfRightSideVec[size
        Sum(prevResidual[split], localPrevResidual[split], prevResidual[split + 1]);
    }

    //  do {
    //      localApproxVec = localApproxVec - methodStep * (localMatrix * approxVec - localRightSideVec);

    //      CalcResidual(localMatrix, approxVec, localRightSideVec, normOfRightSideVec, residual);

    //      ChooseMethodStep(prevResidual, residual);

    //      prevResidual = residual;

    //  } while (CompareResidualAndEpsilon(residual) && iterCount++ < MAX_ITER);
}
```

- В процессе написания аналогичной LuNA программы, старался полностью скопировать код написанный на C++
- Программа работает

Результаты

- Получены навыки программирования с использованием библиотеки MPI
- Создана MPI программа решения СЛАУ методом простой итерации
- Установлен и апробирован инструмент для профилирования MPI программ – TAU
- Написана аналогичная C программе – LuNA программа
- Получены навыки работы на вычислительном кластере общего пользования