XXX Зимняя школа по параллельному программированию 2018

# Разработка библиотеки для параллельного программирования на основе распределенного N-мерного массива

Выполнил: Ижицкий Р. Л.
Руководитель: Городничев М. А.
02.02.2018

# Научные проблемы

- Как организовать повторное использование системного кода

```
for (int i = 0; i < size; ++i)
{
   sendCounts[i] = N / size;
   if (rem > 0)
   {
       sendCounts[i]++;
       rem--;
   }
   displs[i] = sumDispls;
   sumDispls = sumDispls + sendCounts[i];
}
```

- Как облегчить программирование в условиях повышения сложности алгоритмов
- Как преодолеть высокий порог вхождения в область параллельного программирования

# Долгосрочная цель проекта

- Разработка библиотеки для управления распределенным N-мерным массивом

# Требования

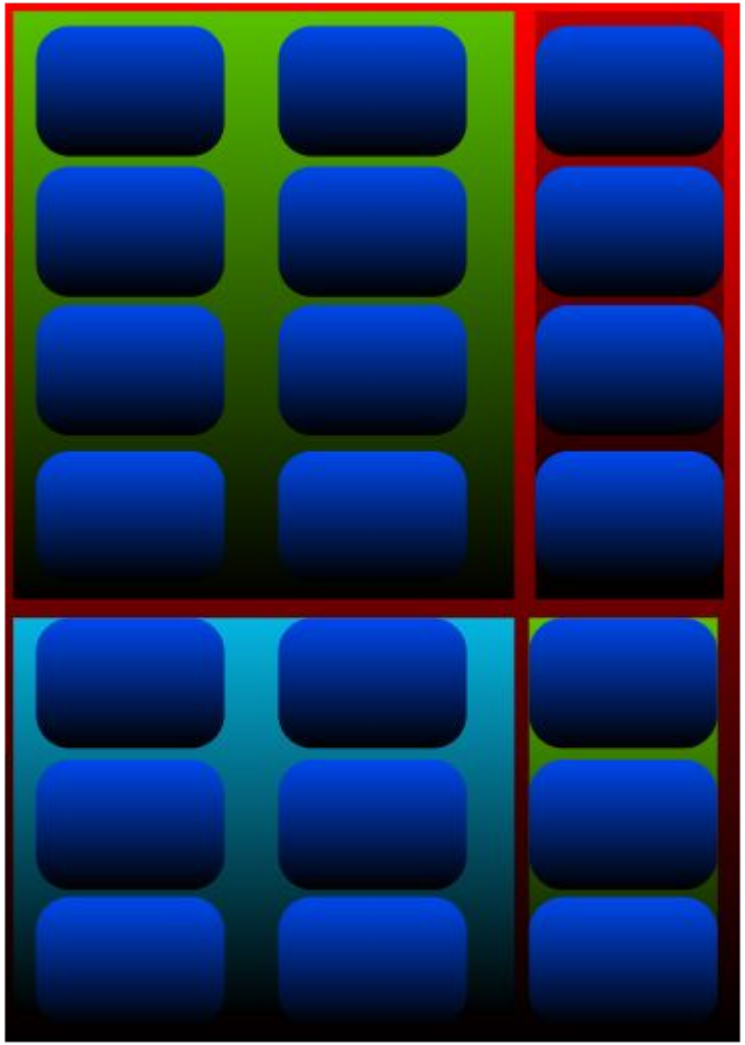
- Удобный и логичный для пользователя интерфейс
- Возможность получения данных с удаленных узлов

# Задачи ЗШ2018

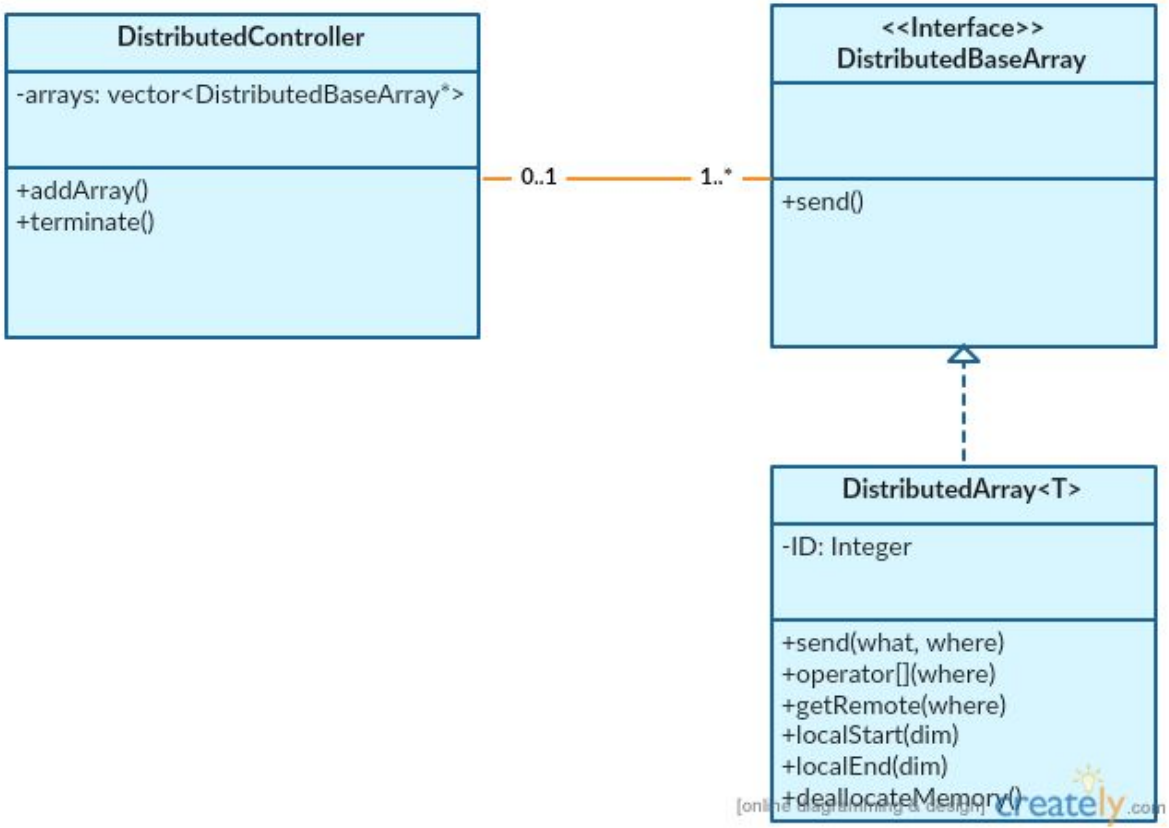
- Отладка синхронного запроса элементов
- Реализация асинхронного обмена границами
- Разработка приложения для решения уравнения Пуассона
- Тестирование библиотеки на этом приложении

# Пример

```
int main(int argc, char *argv[]) {
    DistributedController controller(argc, argv);
    int rank, size;
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    DistributedArray<int> a = DistributedArray<int>(controller, {7, 3}, {2, 2}, {rank
/ 2, rank % 2});
    for (size_t i = a.localStartInDim(0); i < a.localEndInDim(0); i++) {
        for (size_t j = a.localStartInDim(1); j < a.localEndInDim(1); j++) {
            a[{i, j}] = i * 100 + j * 10 + rank;   } }
    int arr[3];
    MPI_Request sendRequest, recRequest;
    if (rank == 1) { a.asyncRequest({4, 0}, 3, sendRequest, recRequest, arr);}
    if (rank == 1) {
        MPI_Wait(&recRequest, MPI_STATUS_IGNORE);
        MPI_Wait(&sendRequest, MPI_STATUS_IGNORE);    }
    a.deallocate();    controller.terminate();
    MPI_Finalize();
    return 0; }
```

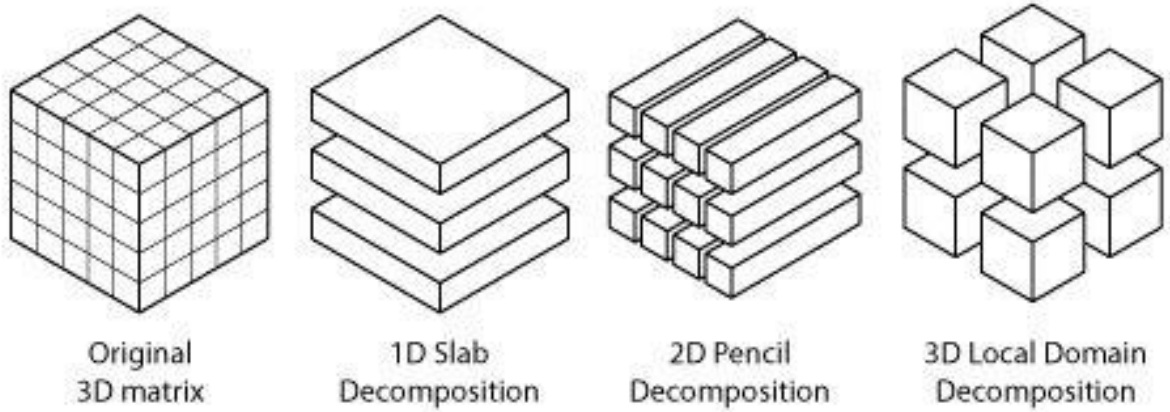


# Реализационные детали



# Уравнение Пуассона в 3D области

<link>: Пример использования библиотеки



# Результаты тестирования

Задача Пуассона с 1D декомпозицией
(в сек) (-O3) (40x40x40):

Исходная версия
2.367870 - 2 threads
4.177221 - 1 thread

С библиотекой
11.821070 - 2 threads
20.881432 - 1 thread

# Заключение

- Разработан прототип библиотеки:
  - Распределенный массив
  - Удобное обращение
  - Асинхронный запрос блока данных

- Протестировано:
  - Уравнение Пуассона в 3D области с 1D декомпозицией

# Планы

- Обмены границами при 2D и 3D декомпозиции данных
  - MPI_Type_vector
- Представление части массива, хранимой в узле, в виде множества фрагментов

```
while (fr = arrA.localFragments.getFragment())
   for (i = fr.localStart(); i < fr.localEnd(); i++)
       fr[i] = foo(i)
```

- Управление распределением фрагментов (в том числе в динамике)
- Оптимизация алгоритма
- Вывод массива в файл

# Спасибо за внимание