POSIX Threads

а)

б)

POSIX – portable operating system interface

Добавление Threads API:

#include <pthread.h>

Компиляция программы:

gcc -o <исполняемый файл> <исходный файл>.c -lpthread

POSIX Threads

Создание потока:

int **pthread_create**(pthread_t *thread, const pthread_attr_t *attr, void *(*start_routine) (void *), void *arg);

Ожидание завершения потока:

int **pthread_join**(pthread_t thread, void **status);

```c
#include <pthread.h>
#include <stdio.h>

void * thread_func(int id)
{
    printf("[Thread] %d\n", id);
    return;
}

void main(int argc, char **argv)
{
    pthread_t  thread1, thread2;

    pthread_create(&thread1, NULL, (void *(*) (void *)) thread_func, (void *) 1);
    pthread_create(&thread2, NULL, (void *(*) (void *)) thread_func, (void *) 2);

    pthread_join(thread1, NULL);
    pthread_join(thread2, NULL);

    return;
}
```

POSIX Threads + MPI

Компиляция программы:

mpicc -o <исполняемый файл> <исходный файл>.c -lpthread

Запуск:

mpirun -n <число процессов>

<исполняемый файл> [аргументы]

```c
#include <pthread.h>
#include "mpi.h"
#include <stdio.h>

int rank, size;
void * thread_func(int id)
{
    printf("[Process]%d [Thread] %d\n", rank, id);
    return;
}

void main(int argc, char **argv)
{
    pthread_t  thread1, thread2;

    /// Initialize Mpi Environment.
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    pthread_create(&thread1, NULL, (void *(*) (void *)) thread_func, (void *) 1);
    pthread_create(&thread2, NULL, (void *(*) (void *)) thread_func, (void *) 2);

    pthread_join(thread1, NULL);
    pthread_join(thread2, NULL);

    MPI_Finalize();
    return;
}
```

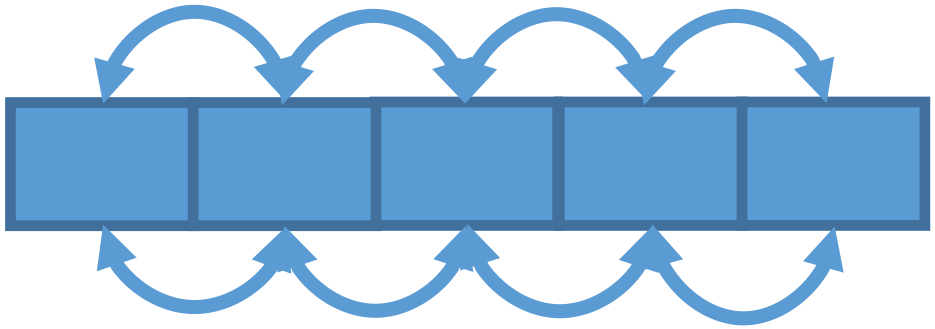
POSIX Threads + MPI

- Потоки довольно просто обмениваются данными по сравнению с процессами.
- Создавать потоки для ОС проще и быстрее, чем создавать процессы.

- Необходима потоковая безопасность функций. Для процессов это не нужно.
- Один бажный поток может повредить остальные. Процессы более изолированы друг от друга.
- Потоки конкурируют друг с другом в адресном пространстве за стек и локальное хранилище.

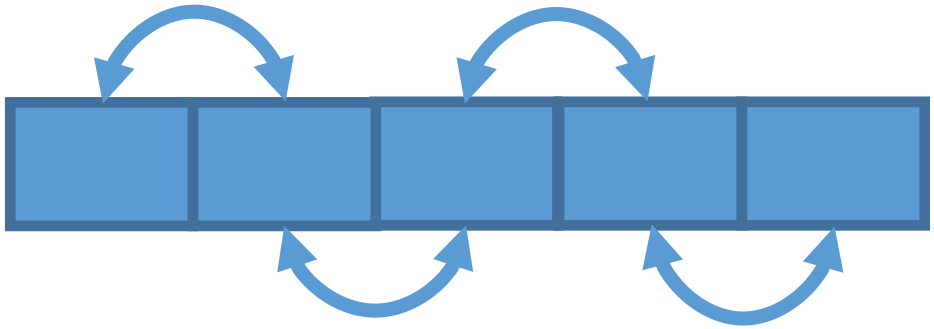
Сортировка

Пузырьковая сортировка



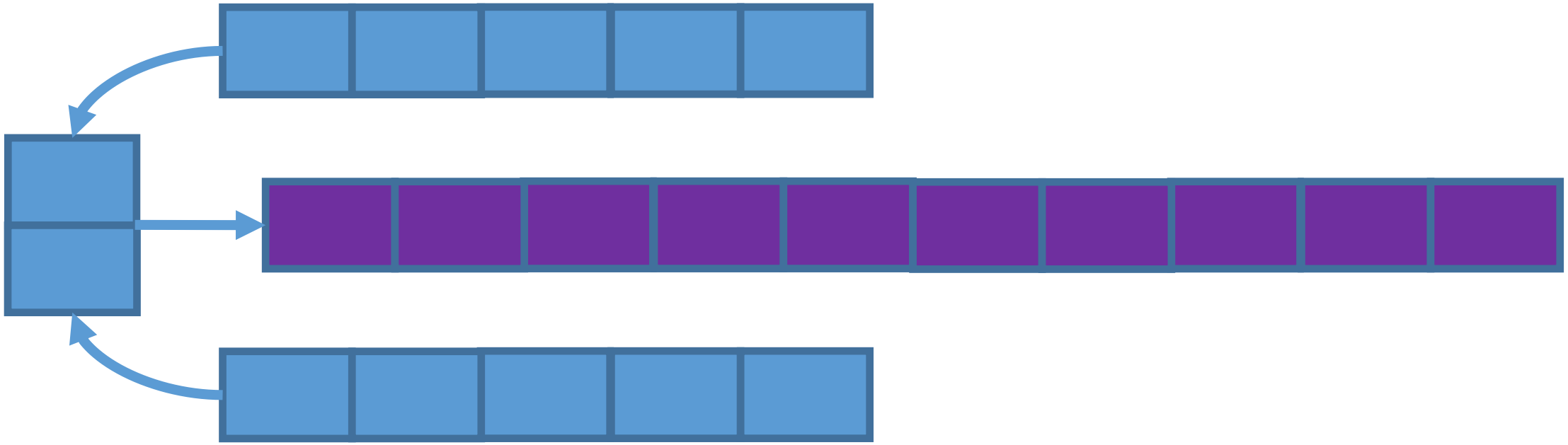
Сортировка

Чёт-нечёт перестановка



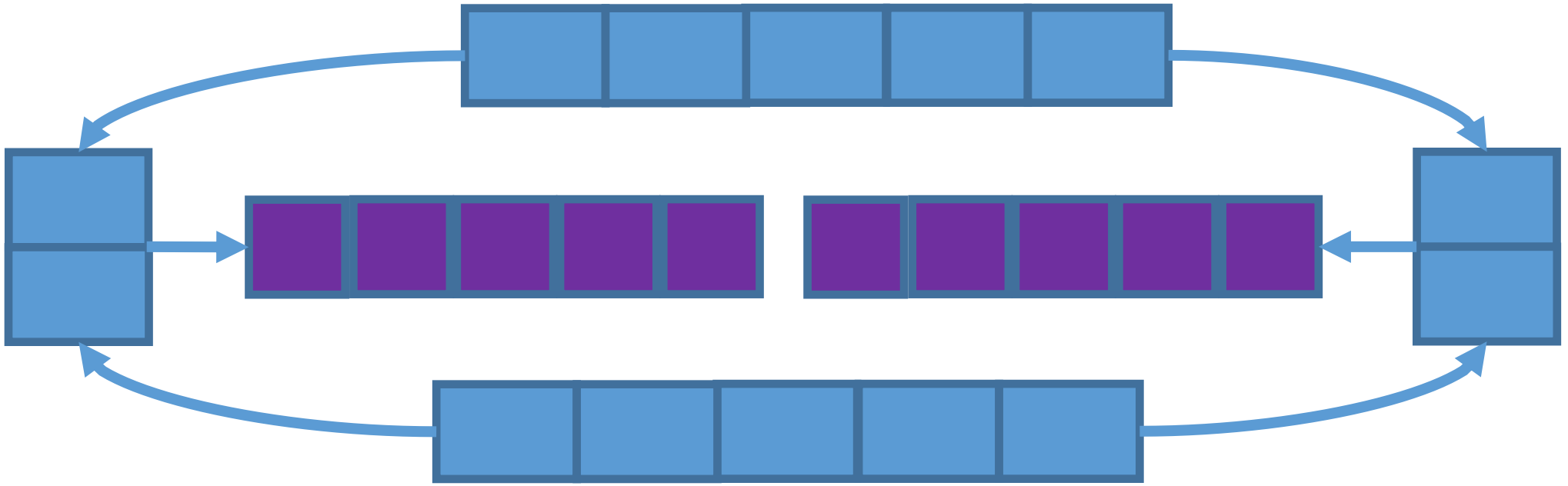
Сортировка

Сортировка слиянием



Сортировка

Сортировка слиянием



Сортировка

Чёт-нечёт слияние