

**Интернет Университет  
Суперкомпьютерных технологий**

**Учебный курс**

***Введение в параллельные алгоритмы***

**Лекция 4**

**Сортировка данных с точки зрения МВС  
(окончание)**

Якобовский М.В., д.ф.-м.н.  
Институт математического  
моделирования РАН, Москва

# ОСНОВНАЯ

---

Расположить в порядке  
неубывания  
 $N$  элементов массива  
чисел,  
используя  $p$  процессоров

ЦЕЛЬ

# Две задачи сортировки массива чисел

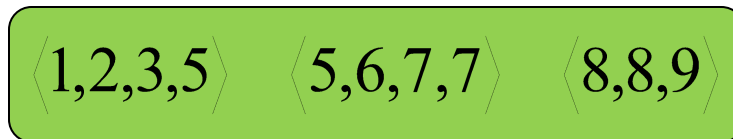
---

- A. Объём оперативной памяти одного процессорного узла **достаточен** для одновременного размещения в ней всех элементов массива
  
- B. Объём оперативной памяти одного процессорного узла **мал** для одновременного размещения в ней всех элементов массива

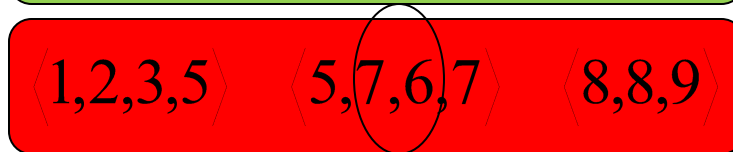
# Задача В

- ❑ Части массива хранятся на нескольких процессорах
  - Каждая часть массива должна быть упорядочена
  - На процессорах с большими номерами должны быть размещены элементы массива с большими значениями

• Правильно



• Ошибка



• Ошибка



$$N = 11$$

$$p = 3$$

# Задача В

---

- Будем рассматривать только процесс упорядочивания элементов:
  - Перед началом сортировки на каждом из процессоров уже есть часть элементов массива
  - После окончания сортировки на каждом из процессоров должно остаться столько элементов, сколько их было в начале (но, это уже могут быть другие элементы, расположенные ранее на других процессорах)

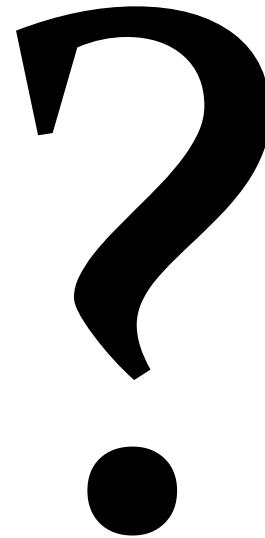
# Этапы сортировки

---

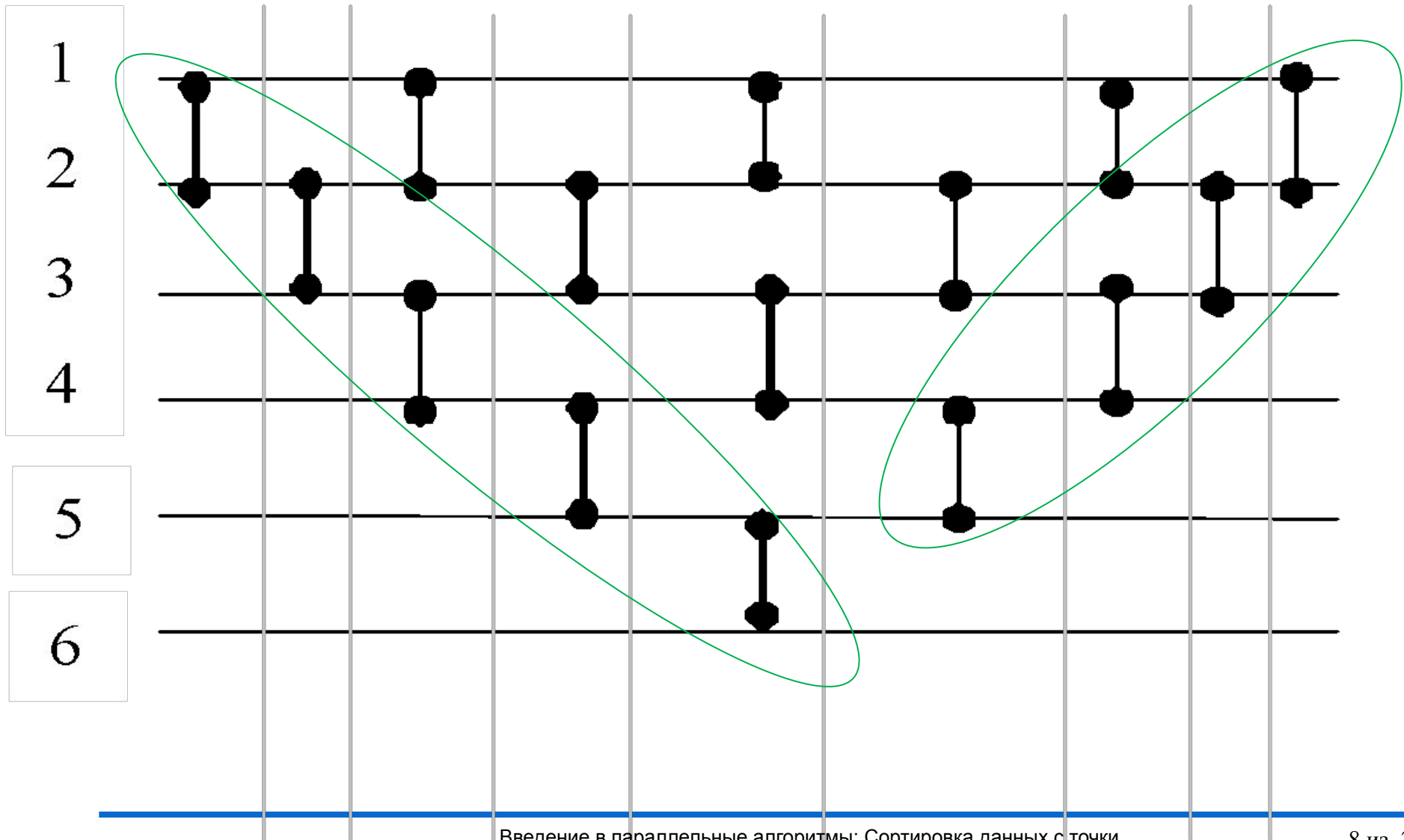
- Упорядочивание фрагментов массива на каждом из процессоров
- Перераспределение элементов массива между процессорами с сохранением упорядоченности массива на каждом отдельном процессоре

# Стратегия перераспределения данных между процессорами

---

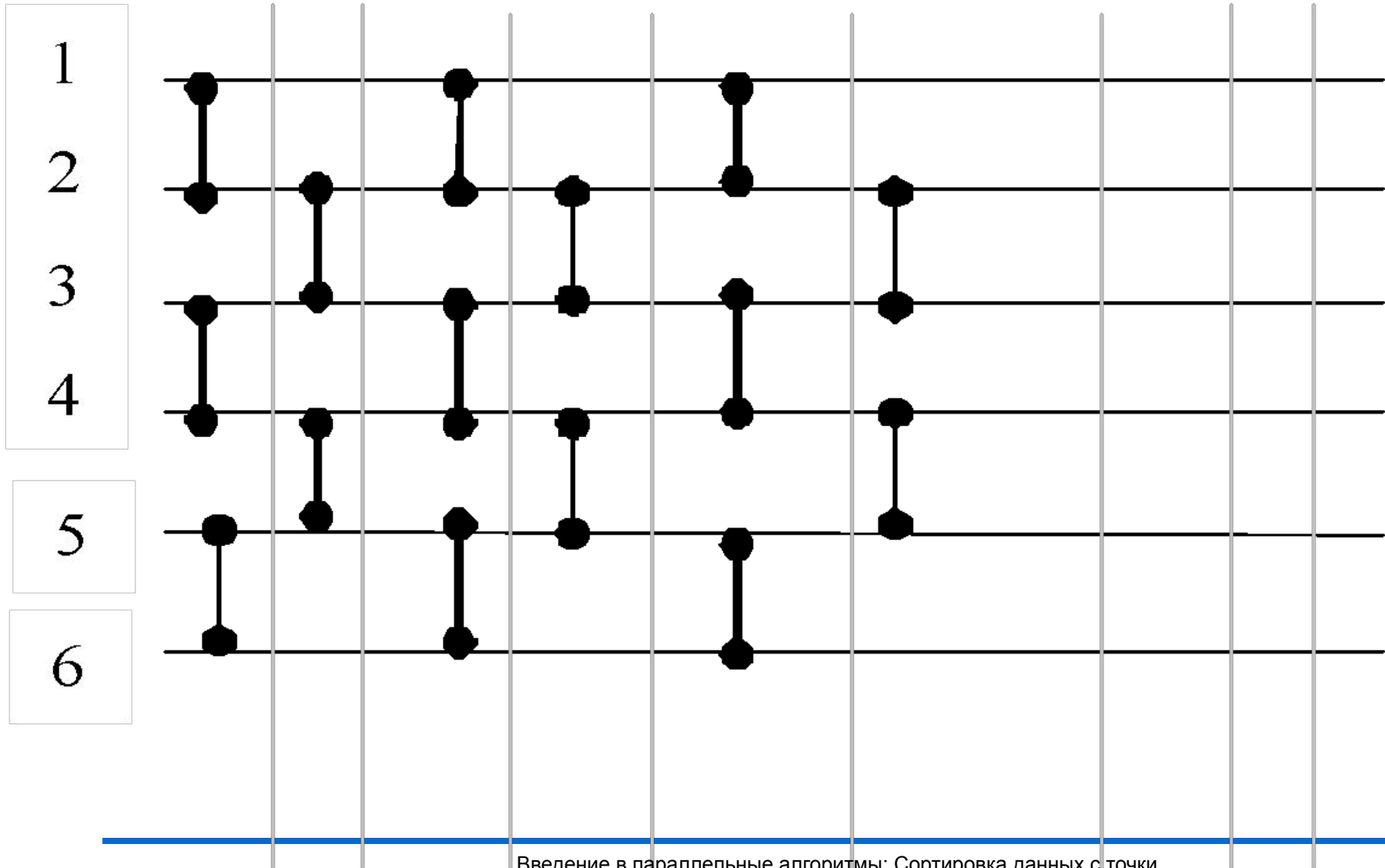


# Сеть сортировки (пузырёк) $n=6$ $s=2n-3=9$

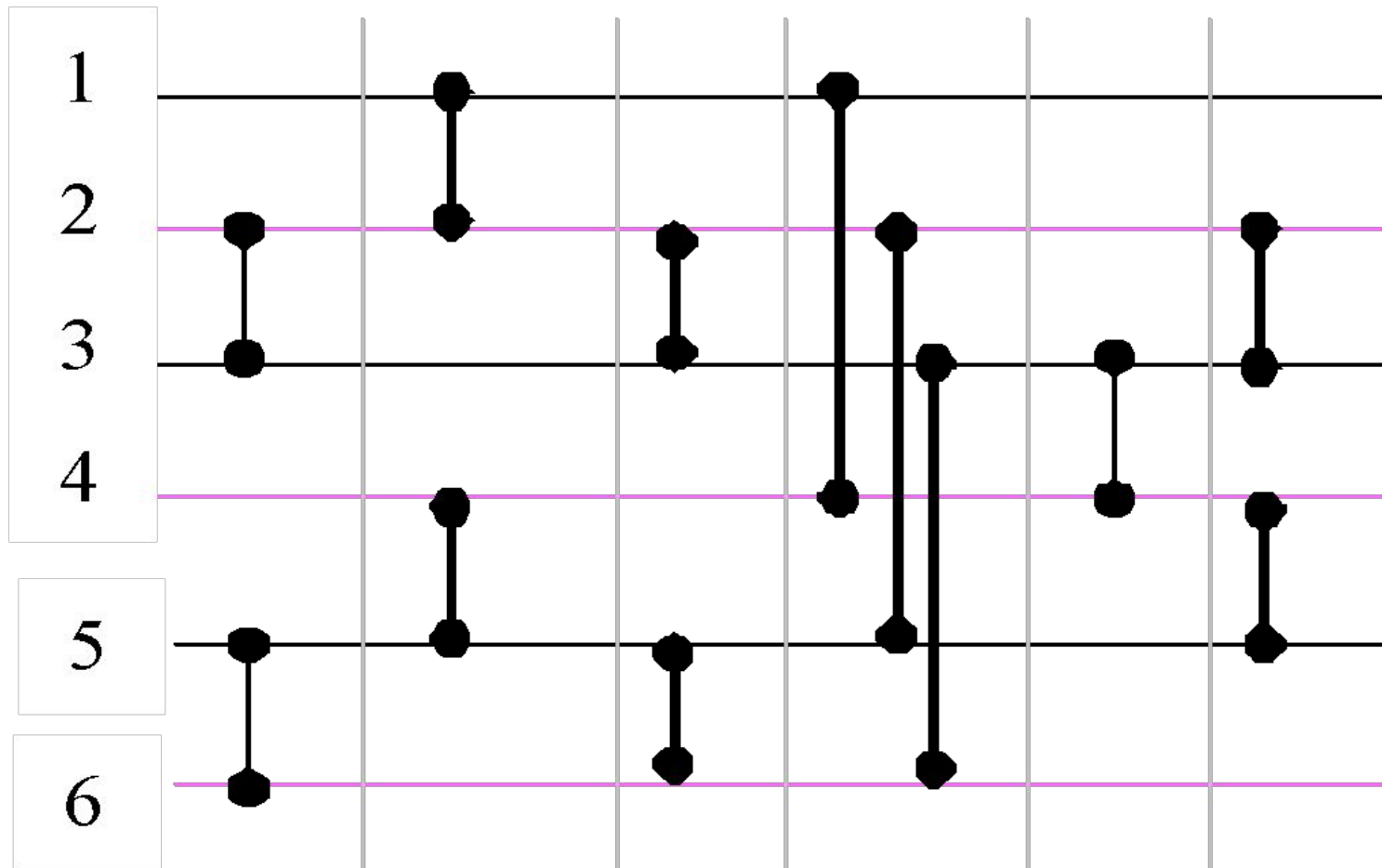




# Сеть сортировки четно-нечетные перестановки $n=6$ $s=n=6$



# Сеть сортировки $n=6$ $s=?=5$



Шаги: 1

2

3

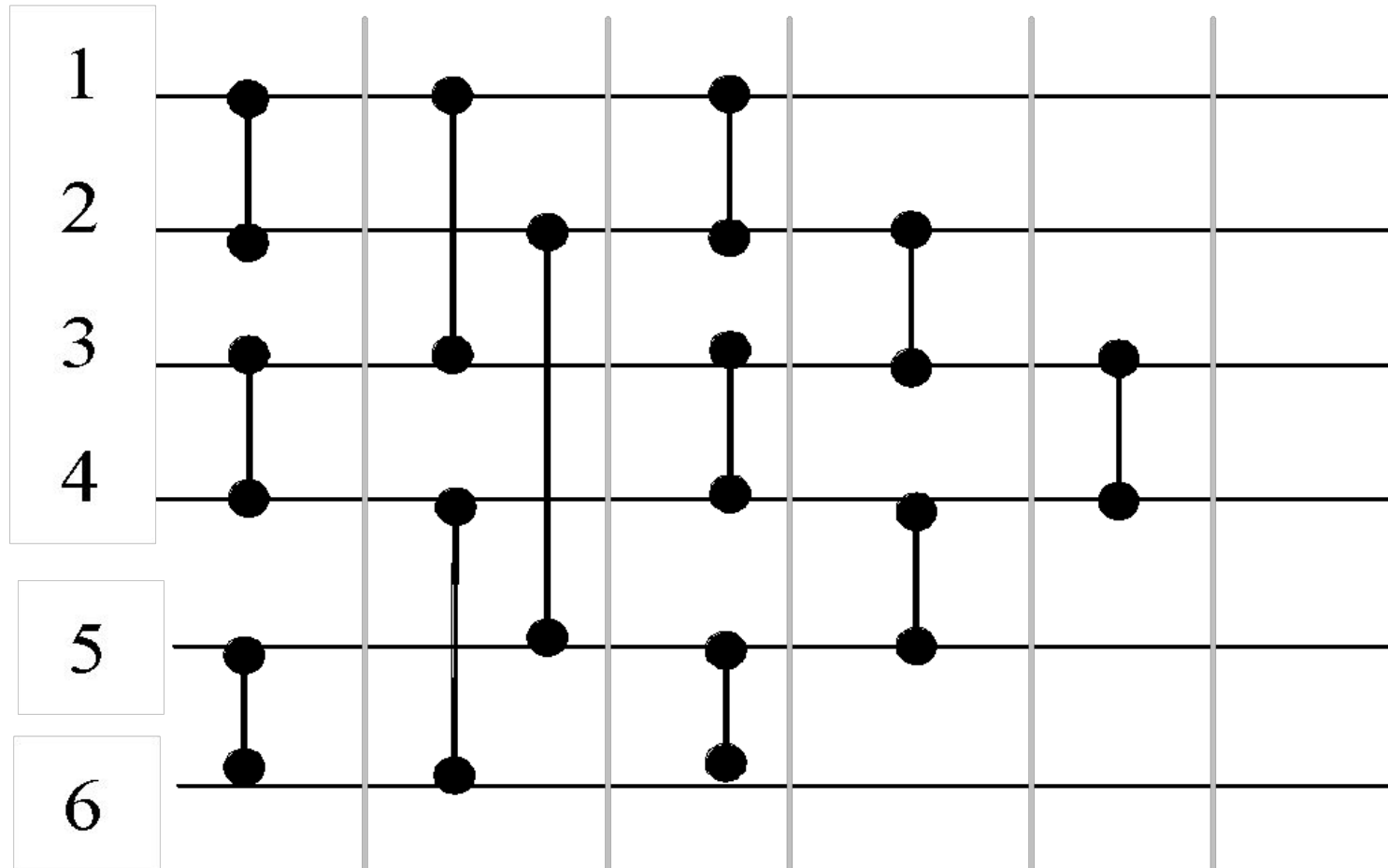
4

5

6

Введение в параллельные алгоритмы: Сортировка данных с точки зрения МВС (окончание) © Якобовский М.В.

# Минимальная сеть сортировки $n=6$ $s=?=5$



Шаги: 1

2

3

4

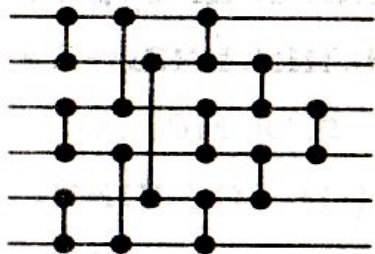
5

Введение в параллельные алгоритмы: Сортировка данных с точки зрения МВС (окончание) © Яковлевский М.В.

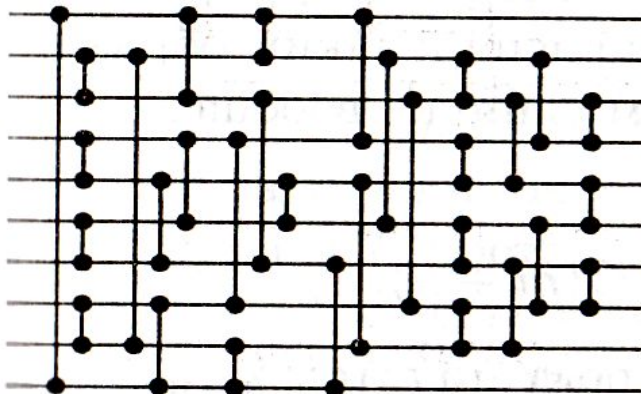
# Минимальные сети сортировки

[Дональд Э.Кнут]

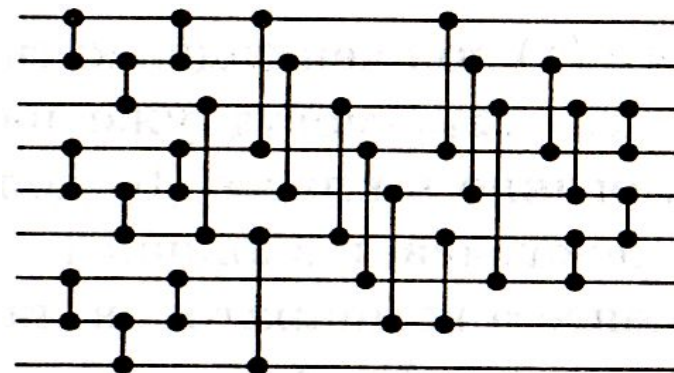
$n=6$   $s=5$



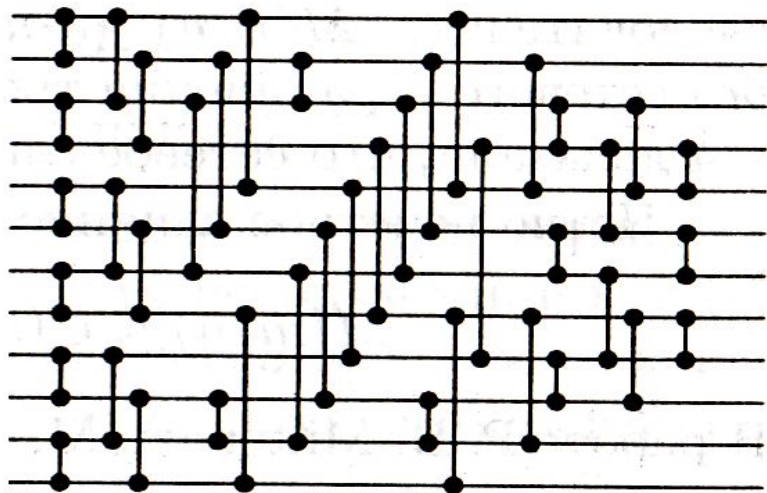
$n=10$   $s=7$



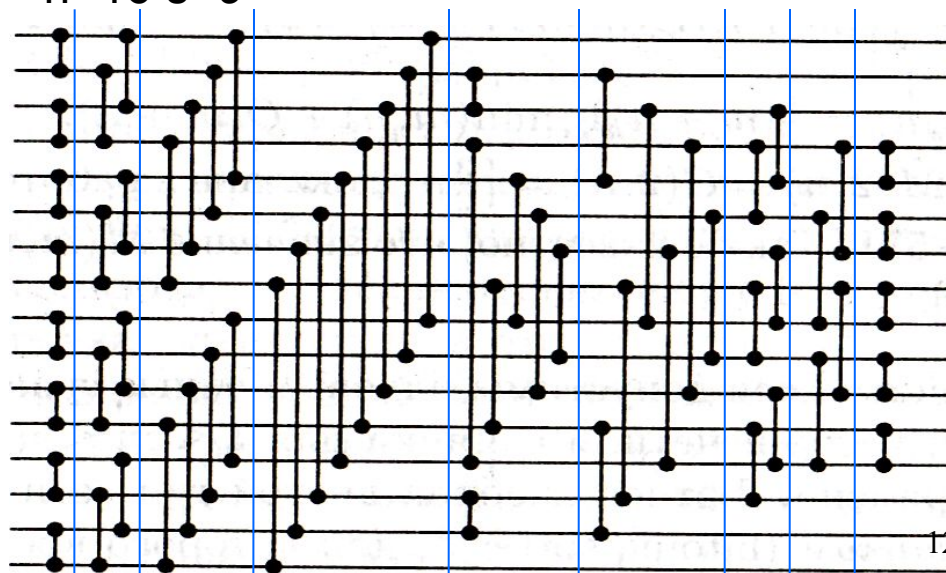
$n=9$   $s=8$



$n=12$   $s=8$



$n=16$   $s=9$



# Четно-нечетное слияние Бэтчера

Объединение двух упорядоченных массивов размера  $m$  и  $n$ :  $\langle x[1] \dots x[m] \rangle$  и  $\langle y[1] \dots y[n] \rangle$ .

- а) Если  $m=0$  или  $n=0$ , то сеть пуста
- б)  $m=n=1$ , то нужен один компаратор.
- с) Если  $mn > 1$ , то

сольём нечетные элементы последовательностей

$\langle x[1], x[3] \dots x[(m+1)/2-1] \rangle$  и  $\langle y[1], y[3] \dots y[(n+1)/2-1] \rangle$ , и получим  
 $\langle v[1], v[2] \dots v[(n+1)/2+(m+1)/2-2] \rangle$

Сольём четные элементы последовательностей

$\langle x[2], x[4] \dots x[m/2] \rangle$  и  $\langle y[2], y[4] \dots y[n/2] \rangle$ , и получим  
 $\langle w[2], w[2] \dots w[n/2+m/2] \rangle$

Применим операции сравнения перестановки к элементам

$(w_1, v_2), (w_2, v_3), (w_3, v_4), \dots$

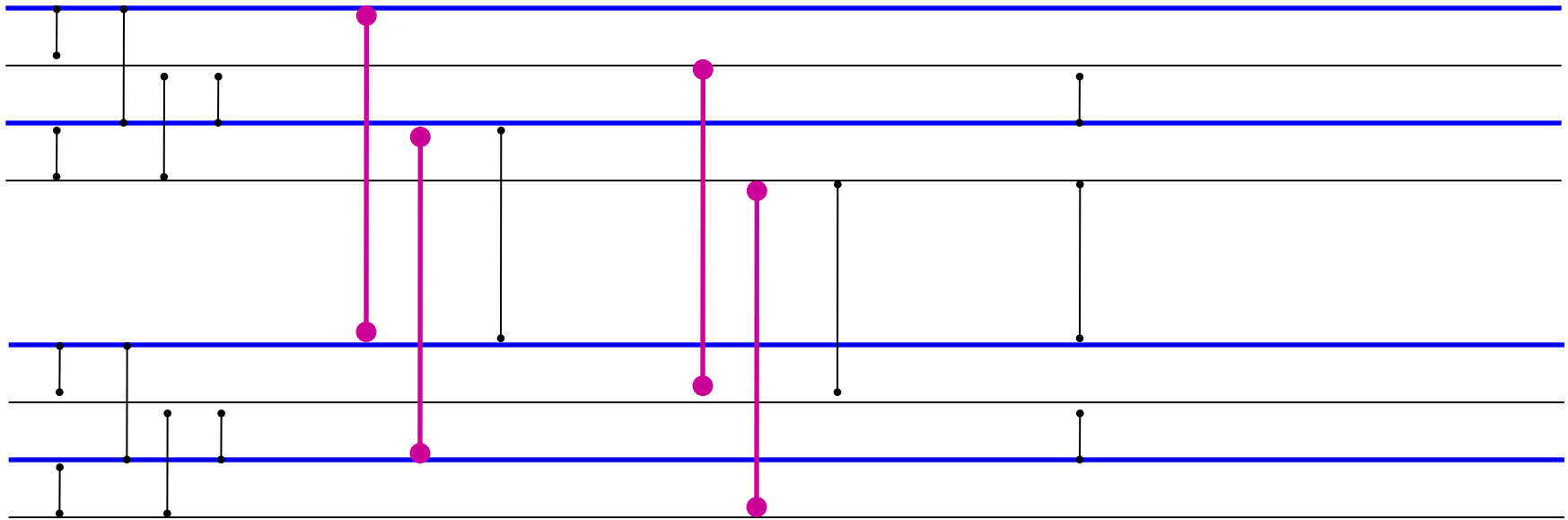
И получим отсортированный массив  $\langle v[1], w[1], v[2], w[3], \dots \rangle$

# Правильность сети

---

- Доказать правильность можно на основе принципа нулей и единиц:
  - Если сеть с  $n$  входами сортирует в порядке неубывания все  $2^n$  последовательности из 0 и 1, то она будет сортировать в том же порядке любую последовательность  $n$  чисел.

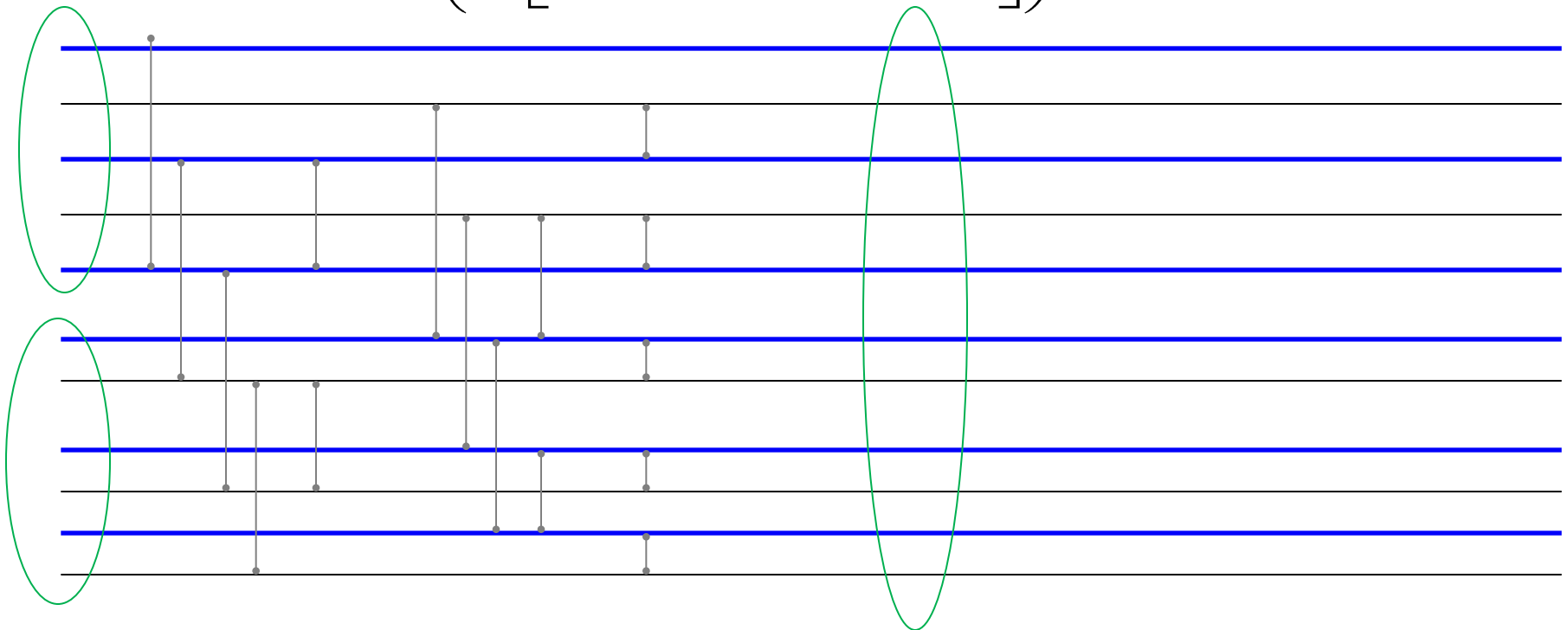
# Сортировка 8ми элементов



# Обменная сортировка со слиянием

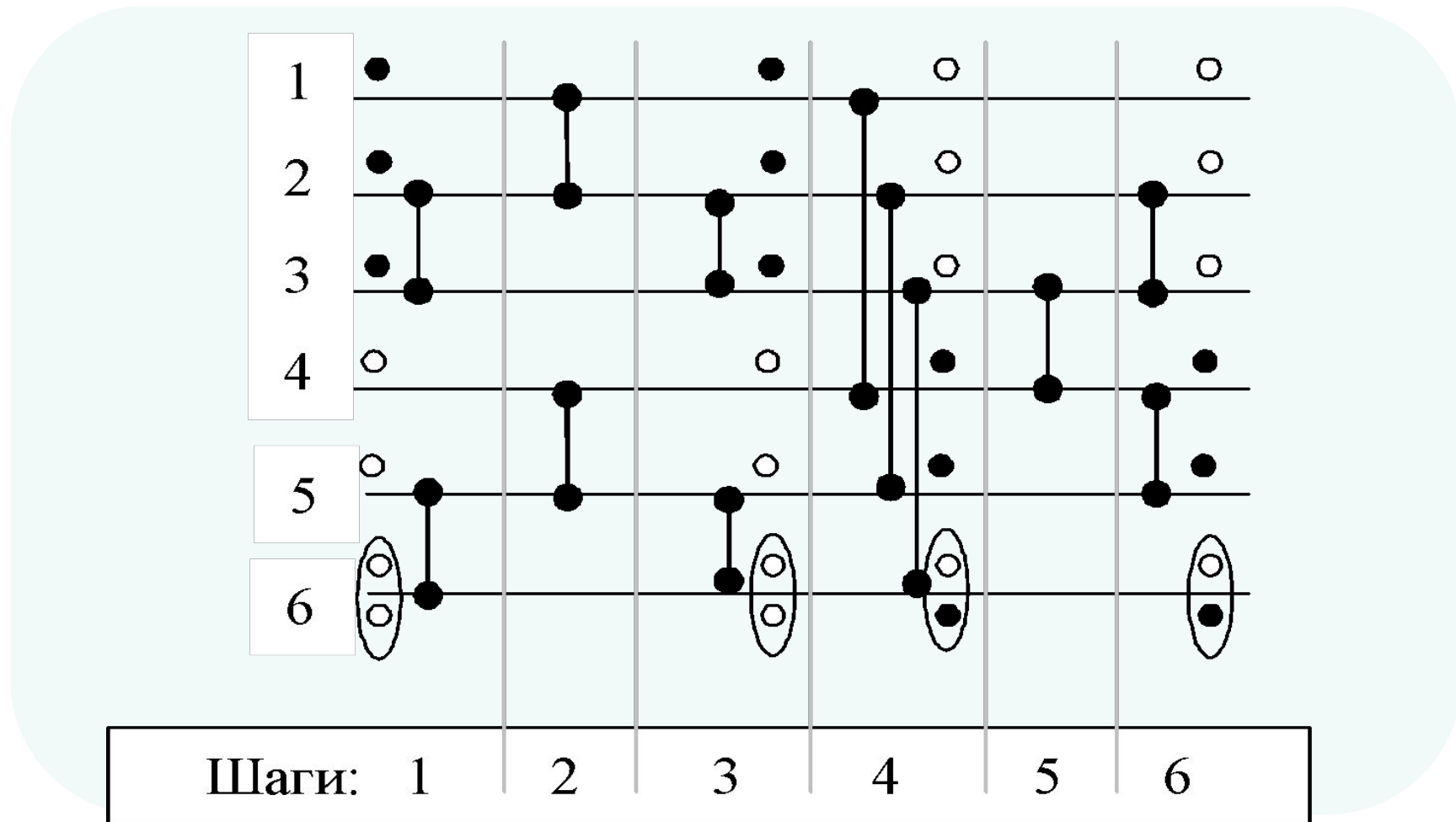
$$O\left(\frac{n}{p} \left[ \log_2 \frac{n}{p} + \frac{\lceil \log_2 p \rceil^2}{2} \right]\right)$$

[Бэтчер]





# Сортировка блоков – ОДИНАКОВОГО РАЗМЕРА



# Сравнение алгоритмов сортировки

Алгоритм сортировки	Среднее число операций
Пирамидальная	$O(n \log_2 n)$
Слияние	$O(n \log_2 n)$
Параллельная сортировка методом сдваивания. Объём данных ограничен оперативной памятью вычислительного узла	$O\left(\frac{n}{p} \log_2 \left(\frac{n}{p}\right) + n\right) \quad T \approx k_c \frac{n}{p} \log_2 \left(\frac{n}{p}\right) + 2k_c n + k_s n$
Параллельная четно-нечетная сортировка Объём данных ограничен общей оперативной памятью	$O\left(\frac{n}{p} \log_2 \frac{n}{p} + n\right) \quad T = k_c \frac{n}{p} \log_2 \left(\frac{n}{p}\right) + (k_c + k_s)n$
Параллельная «обменная сортировка со слиянием»	$O\left(\frac{n}{p} \log_2 \frac{n}{p} + \frac{n}{p} \frac{[\log_2 p]^2}{2}\right)$

# Слияние упорядоченных фрагментов

```
// объединить два упорядоченных массива a, b
```

```
for (ia=0, ib=0, k=0; k<n1+n2; k++)  
{  
    if (ia>=n1) c[k]=b[ib++];  
    else  
        if (ib>=n2) c[k]=a[ia++];  
    else  
        if (a[ia]<b[ib]) c[k]=a[ia++];  
        else c[k]=b[ib++];  
}
```

# Слияние упорядоченных фрагментов

```
for (ia=0, ib=0, k=0; k<n; k++)
```

```
{
```

```
if (ia>=n1) c[k]=b[ib++];
```

```
else
```

```
if (ib>=n2) c[k]=a[ia++];
```

```
else
```

```
if (a[ia]<b[ib]) c[k]=a[ia++];
```

```
else c[k]=b[ib++];
```

```
}
```

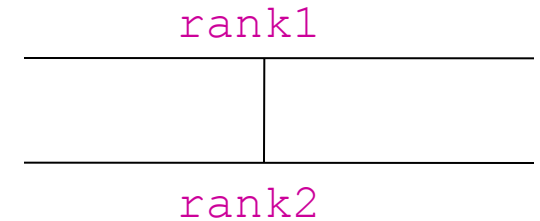
rank1, a[n]

rank2, b[n]

// n – число элементов в каждом из массивов a, b

# Слияние упорядоченных фрагментов

```
Join(int *a, int *b, int *c, int n, rank1, rank2)
{
  if (rank==rank1)
    for (ia=0, ib=0, k=0; k<n;)
      {
        if (a[ia]<b[ib]) c[k++]=a[ia++];
        else           c[k++]=b[ib++];
      }
  else
    for (ia=n-1, ib=n-1, k=n-1; k>=0;)
      {
        if (a[ia]>b[ib]) c[k--]=a[ia--];
        else           c[k--]=b[ib--];
      }
}
```



# Реализация компаратора слияния

```
// взаимодействие процессоров rank и rankC
int *a, *b, *c, *tmp;
ASend(a, n, rankC);
ARecv(b, n, rankC);
ASync();

Join(a, b, c, n, min(rank, rankC), max(rank, rankC));

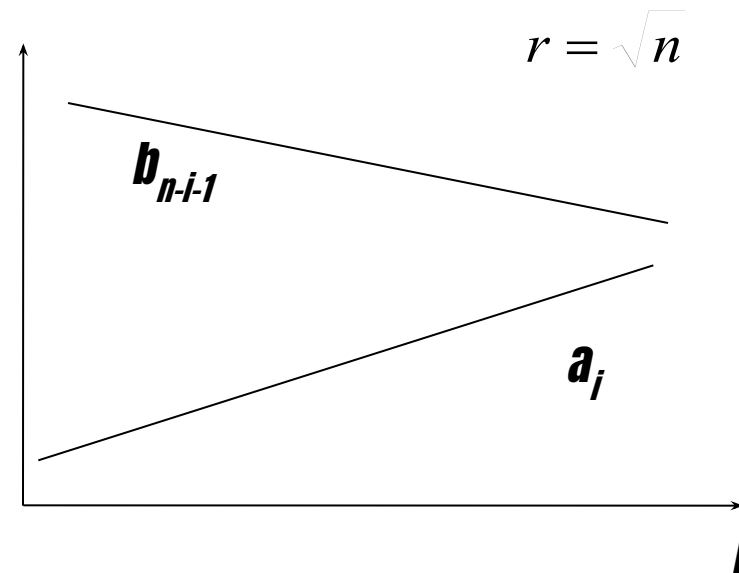
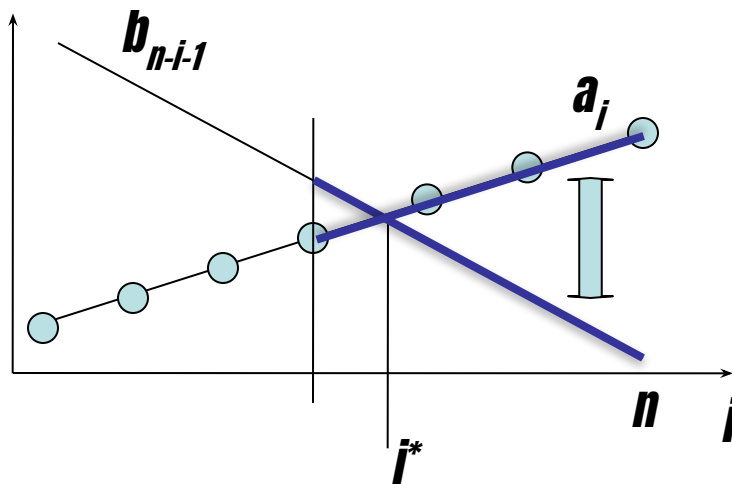
tmp=a;
a=c;
c=tmp;
```

# Сокращение объема передаваемых данных

- Каждое выполнение компаратора слияния требует передачи  $n$  элементов, даже если элементы уже расположены правильно
- На процессоре с меньшим номером подготовим массив, содержащий  $r+1$  элемент массива  $a$  с номерами

$$j \frac{n}{r}, \quad j = 0 \boxtimes n$$

- и передадим его на процессор с большим номером



$n=10^8$ 

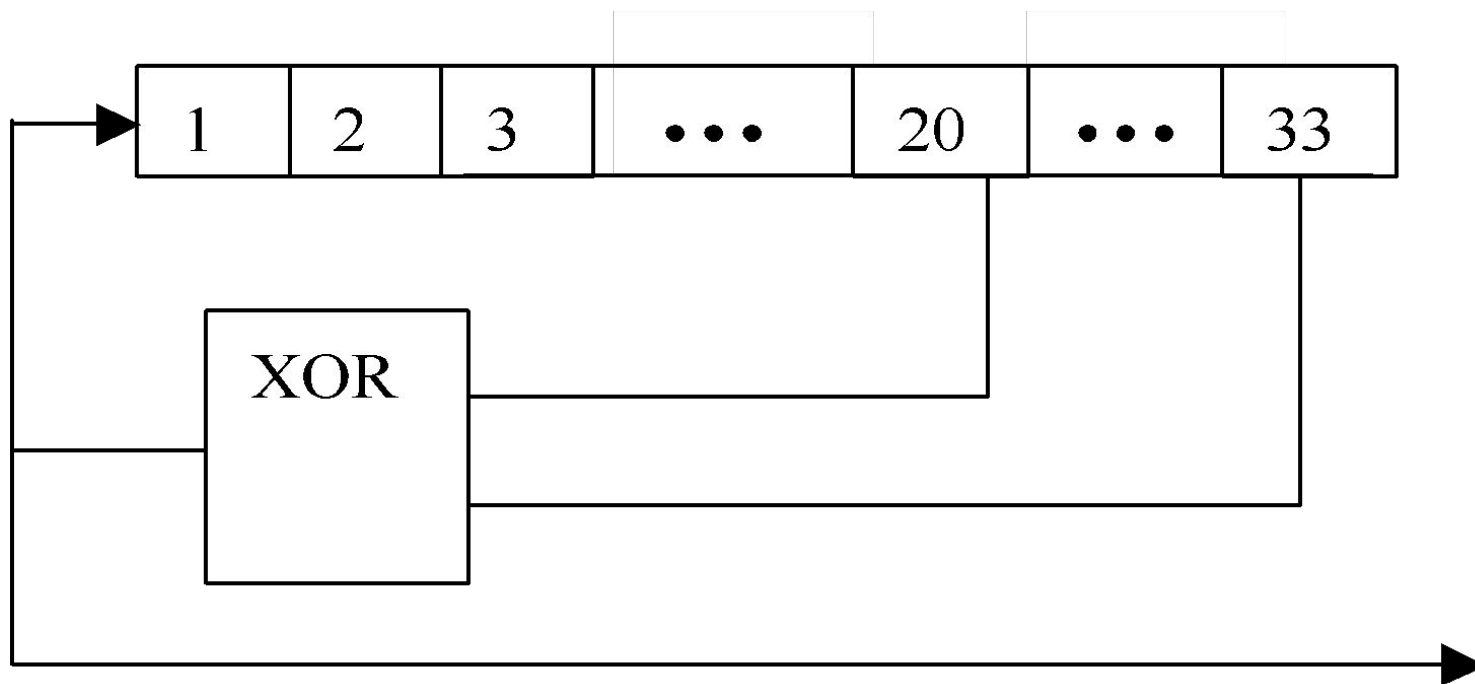
$$E^{\max}(n, p) = \frac{\log_2 n}{\log_2 n + s_p - \log_2 p} \approx \frac{1}{1 + \log_n p (\log_2 p - 1) / 2}$$

<i>P</i>	<i>T,сек</i>	<i>E</i>	<i>S</i>	<i>E<sup>max</sup></i>	<i>S<sup>max</sup></i>	<i>s<sub>p</sub></i>
1	83.51	100.00%	1.00	100%	1.0	0
2	46.40	90.00%	1.80	100%	2.0	1
3	35.93	77.48%	2.32	95%	2.8	3
4	29.68	70.35%	2.81	96%	3.9	3
5	24.45	68.33%	3.42	91%	4.5	5
6	22.16	62.80%	3.77	92%	5.5	5
7	21.82	54.67%	3.83	89%	6.2	6
8	19.95	52.32%	4.19	90%	7.2	6
16	12.36	42.22%	6.75	82%	13.1	10
27	9.32	33.20%	8.97	74%	20.0	14
32	7.85	33.24%	10.64	73%	23.3	15
48	6.45	26.97%	12.95	66%	31.9	19
64	4.92	26.53%	16.98	64%	40.9	21
128	3.19	20.47%	26.20	56%	71.5	28
192	2.52	17.29%	33.19	51%	98.2	33
256	1.99	16.41%	42.02	49%	124.6	36
384	1.63	13.33%	51.20	49%	187.0	41
512	1.29	12.64%	64.74	42%	217.4	45
640	1.21	10.78%	69.02	41%	264.7	47

$$s_p \approx \frac{\lceil \log_2 p \rceil (\lceil \log_2 p \rceil + 1)}{2}$$



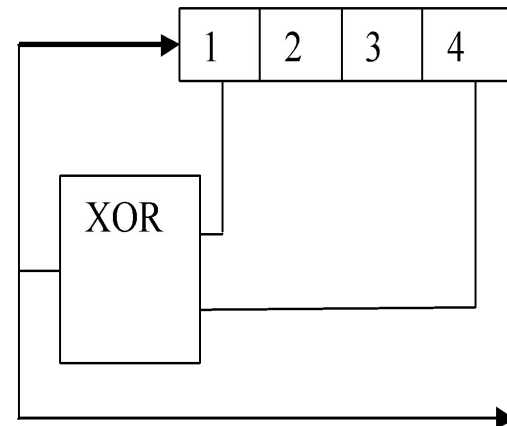
# Генерация псевдослучайных чисел



# Генерация псевдослучайных

ABCD  
 D=C  
 C=B  
 B=A  
 A=(A+D)mod2

	чисел	A	B	C	D	(A+D)%2
1	15	1	1	1	1	0
2	14	0	1	1	1	1
3	13	1	0	1	1	0
4	10	0	1	0	1	1
5	5	1	0	1	0	1
6	11	1	1	0	1	0
7	6	0	1	1	0	0
8	12	0	0	1	1	1
9	9	1	0	0	1	0
10	2	0	1	0	0	0
11	4	0	0	1	0	0
12	8	0	0	0	1	1
13	1	1	0	0	0	1
14	3	1	1	0	0	1
15	7	1	1	1	0	1
16	15	1	1	1	1	0



# Заключение

---

- ❑ Рассмотрен ряд, основанных на сетях методов параллельной сортировки массивов
- ❑ Рассмотрен вопрос оптимизации выполнения слияния на компараторе слияния
- ❑ Рассмотрен вопрос сокращения числа шагов и объема передаваемых данных

# Список литературы

---

1. *Дональд Э.Кнут*. Искусство программирования, т.3. Сортировка и поиск 2-е изд.: Пер. с английского – М.: Издательский дом «Вильямс», 2001.
2. *Седжвик Роберт*. Фундаментальные алгоритмы на C++. Анализ/Структуры данных/Сортировка/Поиск: Пер. с англ./Роберт Седжвик. - СПб.: ООО "ДиаСофтЮП", 2002.-688с.
3. *Якобовский М.В.* Параллельные алгоритмы сортировки больших объемов данных. Москва, 2008,  
<http://lira.imamod.ru/FondProgramm/Sort/ParallelSort.pdf>

**Якобовский М.В.**

д.ф.-м.н.,

зав. сектором

«Программного обеспечения многопроцессорных систем и вычислительных сетей»

Института математического моделирования

Российской академии наук

mail: [mail: lira@imamod.ru](mailto:lira@imamod.ru)

web: <http://lira.imamod.ru>