Параллельные алгоритмы 20230214

Базовые параллельные алгоритмы

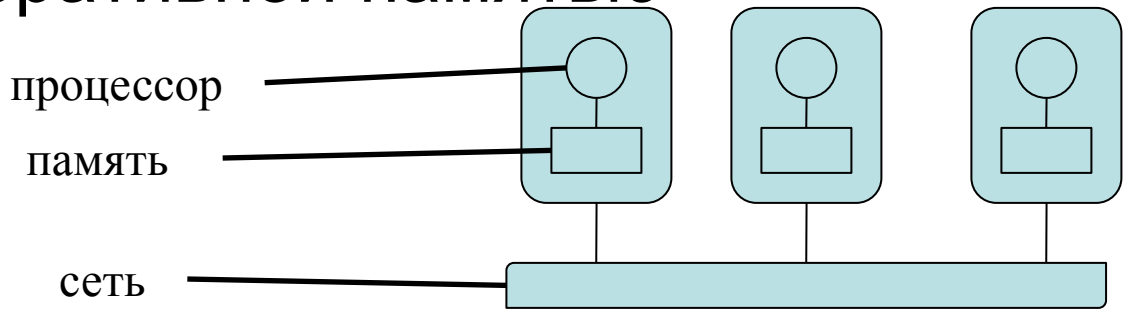
Якобовский Михаил Владимирович

МФТИ, г. Долгопрудный

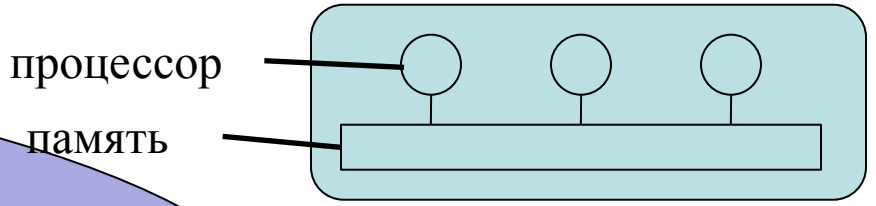
весна, 2023

Круг рассматриваемых систем

- Системы на основе объединенных сетью типовых вычислительных узлов – системы с распределенной оперативной памятью



- Системы с доступом всех процессоров к общей оперативной памяти

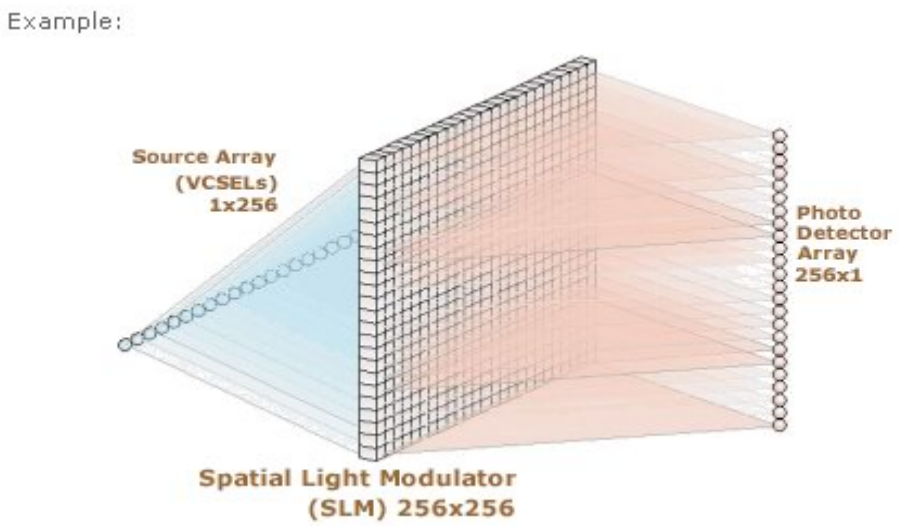


Опер
- Графические ускорители
- ПЛИС
- Нейрокомпьютеры
- Другие …

Оптический векторно-матричный умножитель
$8 \cdot 10^{12}$ операций умножения и сложения в секунду (8TMips)

Оптический процессор Enlight256, 2003 год

10^9 – гига
10^{12} – тера
10^{15} – пета
10^{18} – экса





Энергопотребление процессора составляет 40 Вт (5 мВт на 1 млрд. операций в секунду)

Д.Колисниченко. Оптические процессоры. /
URL: http://dkws.narod.ru/linux/etc/optical/cpu.html

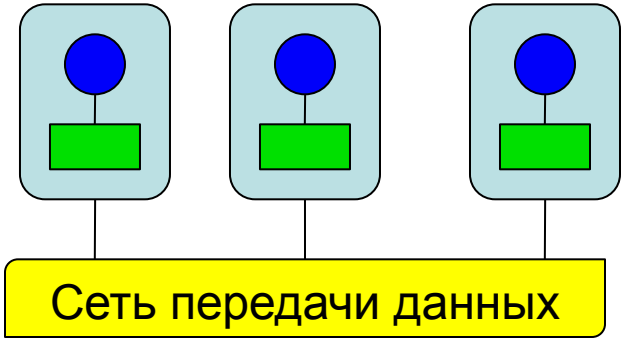
Основные характеристики параллельной программы

- Ускорение $S_p = \frac{T_1}{T_p}$
- Эффективность $E_p = \frac{S_p}{p}$
- Предел масштабируемости – минимальное число процессоров при котором достигается максимальное ускорение

  – Число выполняемых операций
  – Время выполнения
  – Объём обрабатываемых данных

Вычислительные системы с распределённой оперативной памятью

- Системы на основе объединенных сетью типовых вычислительных узлов – системы с распределенной оперативной памятью

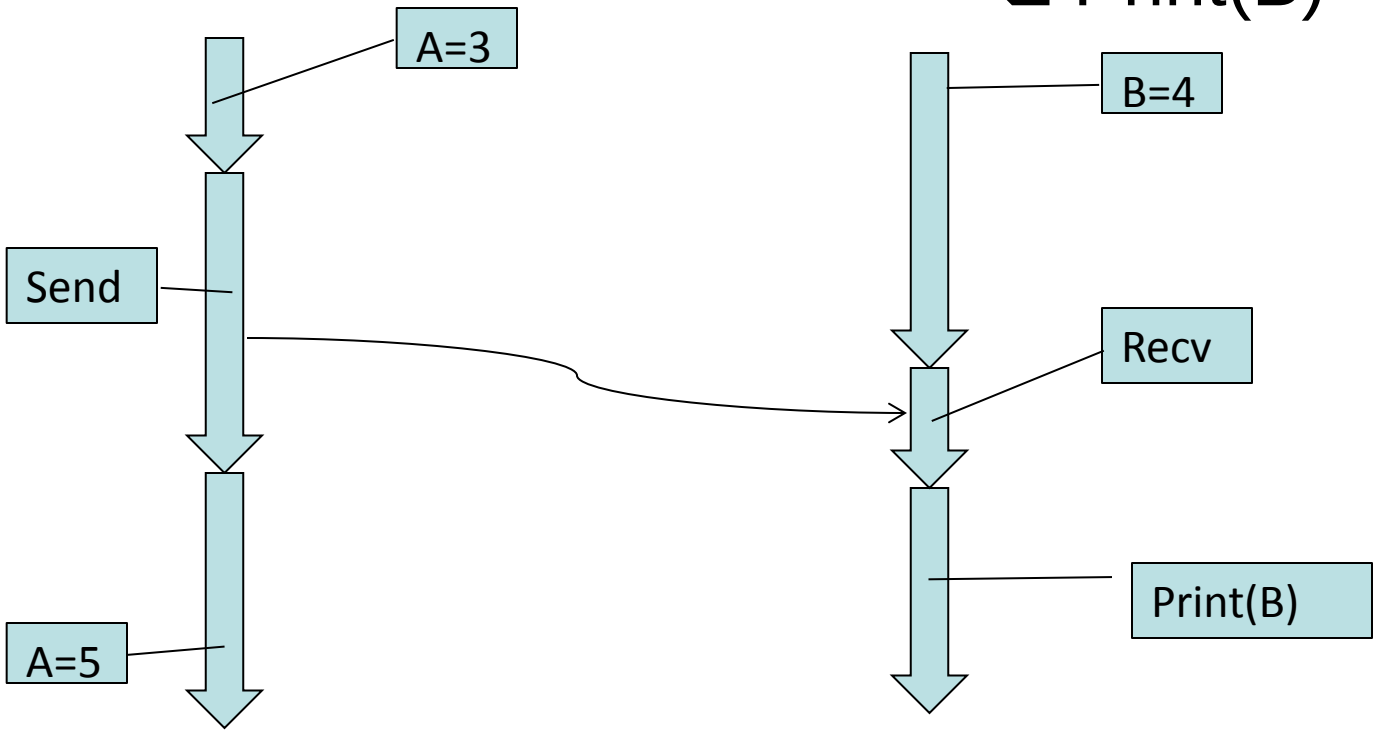




Синхронный метод передачи сообщений

- A=3
- Send(&A)
- A=5

- B=4
- Recv(&B)
- Print(B)



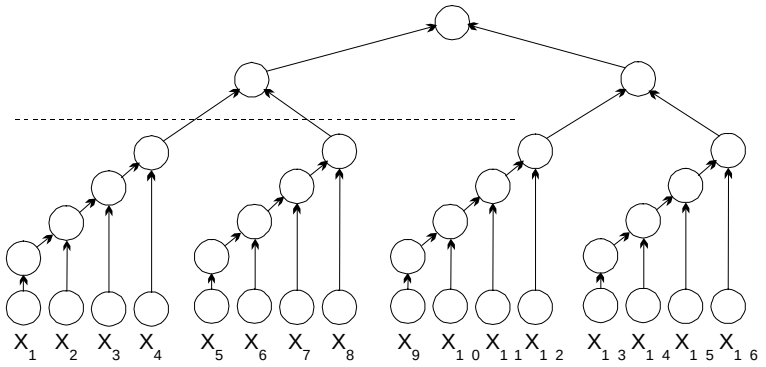
Результат

3

Метод сдваивания

Определение суммы элементов массива

rank	0	1	2	3	4	5	6	7
i	0 … n/8-1	n/8 … 2 n/8-1	2n/8 … 3n/8-1	3n/8 … 4n/8-1	4n/8 … 5n/8-1	5n/8 … 6n/8-1	6n/8 … 7n/8-1	7n/8 … 8n/8-1
	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])
1	Recv(1,c)	Send(0,s)	Recv(3,c)	Send(2,s)	Recv(5,c)	Send(4,s)	Recv(7,c)	Send(6,s)
	s = s + c		s = s + c		s = s + c		s = s + c	
2	Recv(2,c)		Send(0,s)		Recv(6,c)		Send(4,s)	
	s = s + c				s = s + c			
3	Recv(4,c)				Send(0,s)			
	s = s + c							



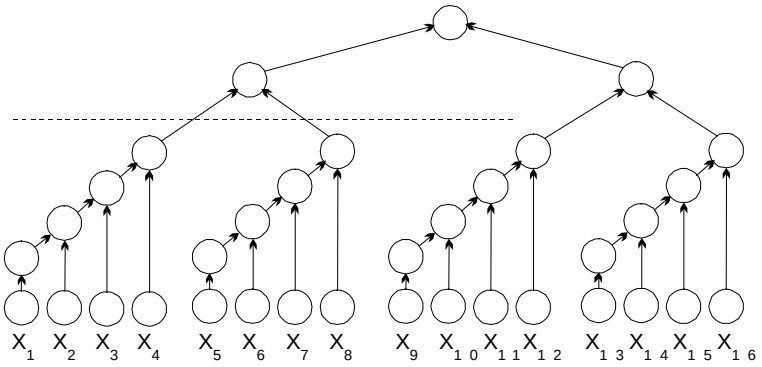
$$T_p = \frac{n}{p}\tau_c + (\tau_c + \tau_s)\log p$$

$$S_p = p\frac{1}{1 + \left(1 + \frac{\tau_s}{\tau_c}\right)\frac{p}{n}\log p}$$

Метод сдваивания

Определение суммы элементов массива

rank	0	1	2	3	4	5	6	7
i	0 … n/8-1	n/8 … 2 n/8-1	2n/8 … 3n/8-1	3n/8 … 4n/8-1	4n/8 … 5n/8-1	5n/8 … 6n/8-1	6n/8 … 7n/8-1	7n/8 … 8n/8-1
	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])
1	Recv(1,c) ←	Send(0,s)	Recv(3,c) ←	Send(2,s)	Recv(5,c) ←	Send(4,s)	Recv(7,c) ←	Send(6,s)
	s = s + c		s = s + c		s = s + c		s = s + c	
2	Recv(2,c) ←		Send(0,s)		Recv(6,c) ←		Send(4,s)	
	s = s + c				s = s + c			
3	Recv(4,c) ←				Send(0,s)			
	s = s + c							



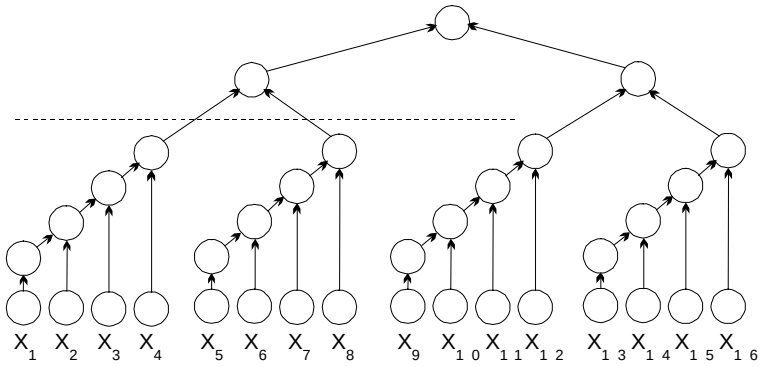
$$T_p = \frac{n}{p}\tau_c + (\tau_c + \tau_s)\log p$$

$$S_p = p\frac{1}{1 + \left(1 + \frac{\tau_s}{\tau_c}\right)\frac{p}{n}\log p}$$

Метод сдваивания

Определение суммы элементов массива

rank	0	1	2	3	4	5	6	7
i	0 … n/8-1	n/8 … 2 n/8-1	2n/8 … 3n/8-1	3n/8 … 4n/8-1	4n/8 … 5n/8-1	5n/8 … 6n/8-1	6n/8 … 7n/8-1	7n/8 … 8n/8-1
	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])	s = sum(a[i])
1	Recv(1,c)	Send(0,s)	Recv(3,c)	Send(2,s)	Recv(5,c)	Send(4,s)	Recv(7,c)	Send(6,s)
	s = s + c		s = s + c		s = s + c		s = s + c	
2	Recv(2,c)		Send(0,s)		Recv(6,c)		Send(4,s)	
	s = s + c				s = s + c			
3	Recv(4,c)				Send(0,s)			
	s = s + c							

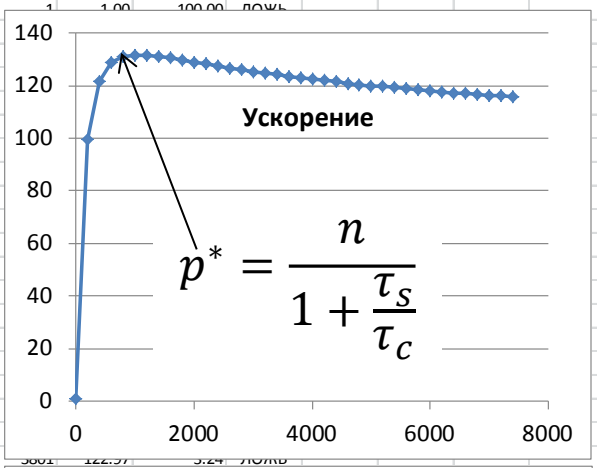


$$T_p = \frac{n}{p}\tau_c + (\tau_c + \tau_s)\log p$$

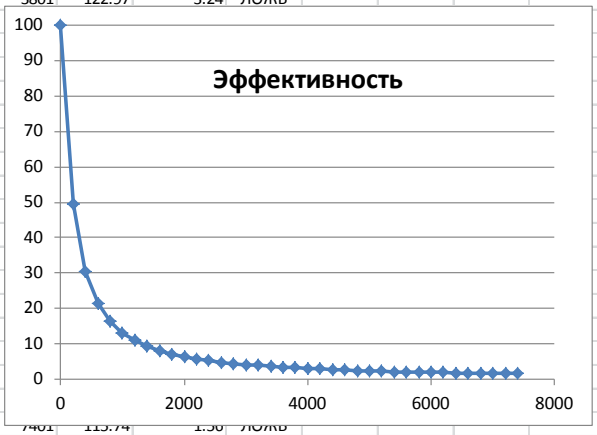
$$S_p = p\frac{1}{1 + \left(1 + \frac{\tau_s}{\tau_c}\right)\frac{p}{n}\log p}$$

Масштабируемость – число процессоров, при котором достигается максимальное ускорение

ts/tc	1 000															
n=2^c	1 048 576	20														
step	200													10 000		
p*	1 047.53															
log(p)	p	S	E										log(n)	n	S*	p*
0.00																20
5.30																30
5.99																40
6.40																50
6.69																60
6.91																70
7.09																80
7.24																90
7.38																100
7.50																110
7.60																120
7.70																130
7.78																140
7.86																150
7.94																160
8.01																170
8.07																180
8.13																190
8.19																200
8.24																210
8.29																220
8.34																230
8.39																240
8.43																250
8.48																260
8.52																270
8.56																280
8.59																290
8.63																300
8.67																310
8.70																320
8.73																330
8.76																340
8.79																350
8.82																360
8.85																370
8.88																380
8.91																390









Неубедительность тестирования

… если для нас представляют интерес реально работающие системы, то требуется убедиться (и убедить всех сомневающихся) в корректности наших построений

… системе часто придется работать в невоспроизводимых обстоятельствах, и мы едва ли можем ожидать сколько-нибудь серьезной помощи от тестов

Dijkstra E.W.

1966

Метод геометрического параллелизма

Циклическая обработка локально связанных данных

- Обработка изображений
- Обработка данных, заданных на решетках или произвольных графах
- Моделирование физических процессов (течений жидкости и газов, теплопереноса, упругости, …)
- …

Одномерная явная разностная схема для уравнения диффузии

$$N_t = D\frac{\partial^2 N}{\partial x^2}$$

$$\frac{\overline{N}_i - N_i}{\tau} = D\frac{N_{i+1} - 2N_i + N_{i-1}}{h^2}, \quad i = 1, \ldots, n-1$$

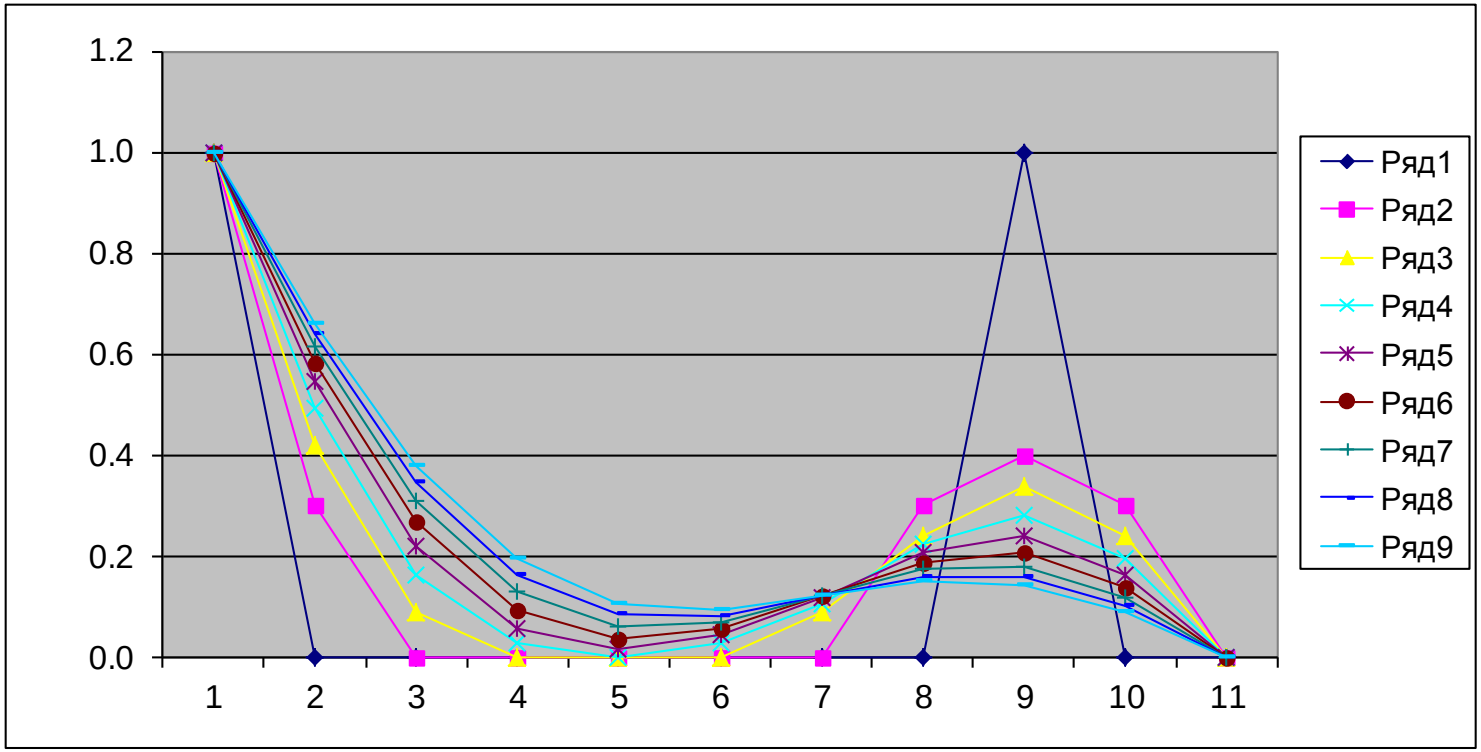
$$\overline{N}_0 = N_0 \quad \overline{N_n} = N_n$$

n – число ячеек

κ – число шагов по времени

$$\overline{N}_i = N_i + \tau\frac{D}{h^2}(N_{i+1} - 2N_i + N_{i-1}), \qquad i = 1, \ldots, n-1$$

Решения одномерного уравнения, соответствующие 9ти моментам времени



Одномерная явная разностная схема для уравнения диффузии

n – число ячеек

κ – число шагов по времени

$$for(j = 0; j < k; j++)$$

{

$$for(i = 1; i < n; i++)$$

$$\overline{N}_i = N_i + \tau \frac{D}{h^2}(N_{i+1} - 2N_i + N_{i-1})$$

$$\overline{N}_0 = N_0$$

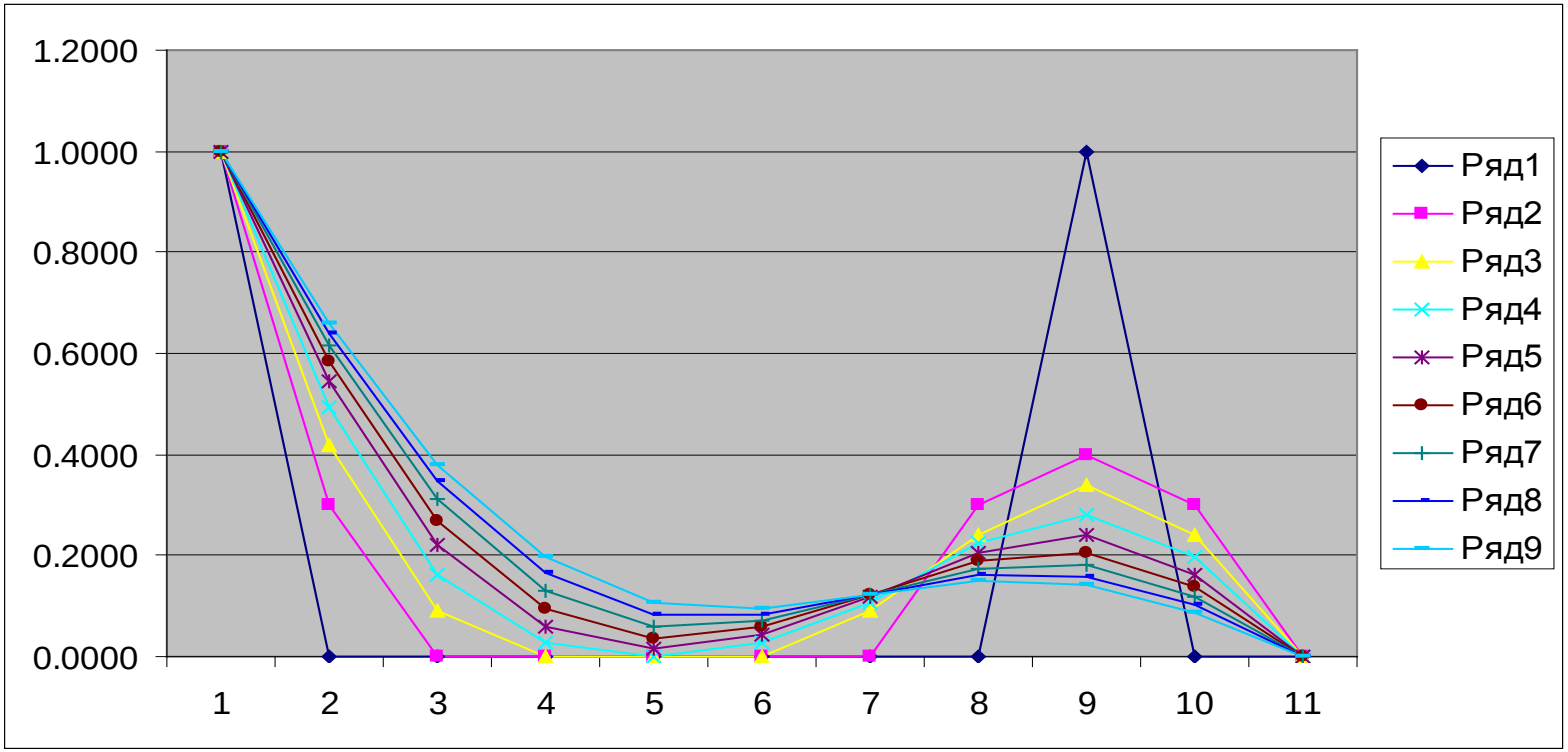
$$\overline{N}_n = N_n$$

$$for(i = 0; i \le n; i++)$$

$$N_i = \overline{N_i}$$

}

Решения одномерного уравнения, соответствующие 9ти моментам времени

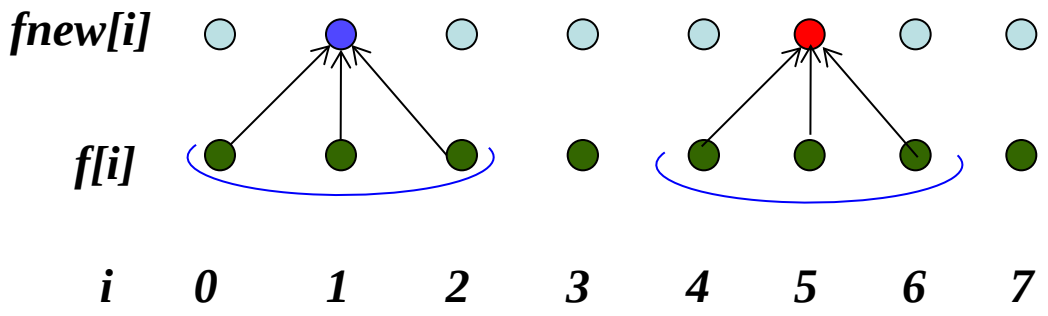


Метод геометрического параллелизма

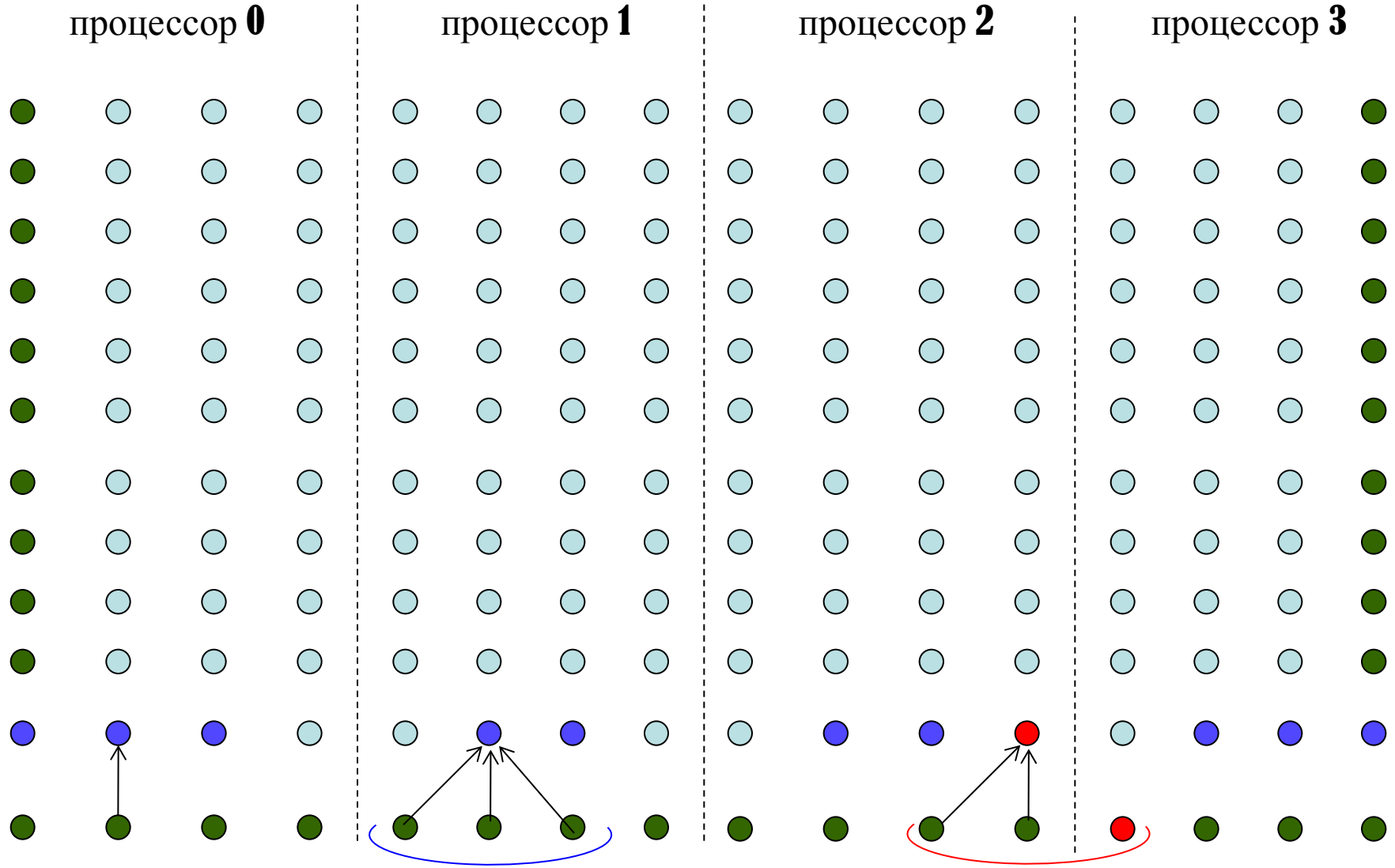
```
for(t=0; t<k; t++)
{
    for( i=1; i<n; i++)
        fnew[i]= fun( f[i-1], f[i], f[i+1] )

    for( i=0; i<=n ; i++)
        f[i] = fnew [i]
}
```

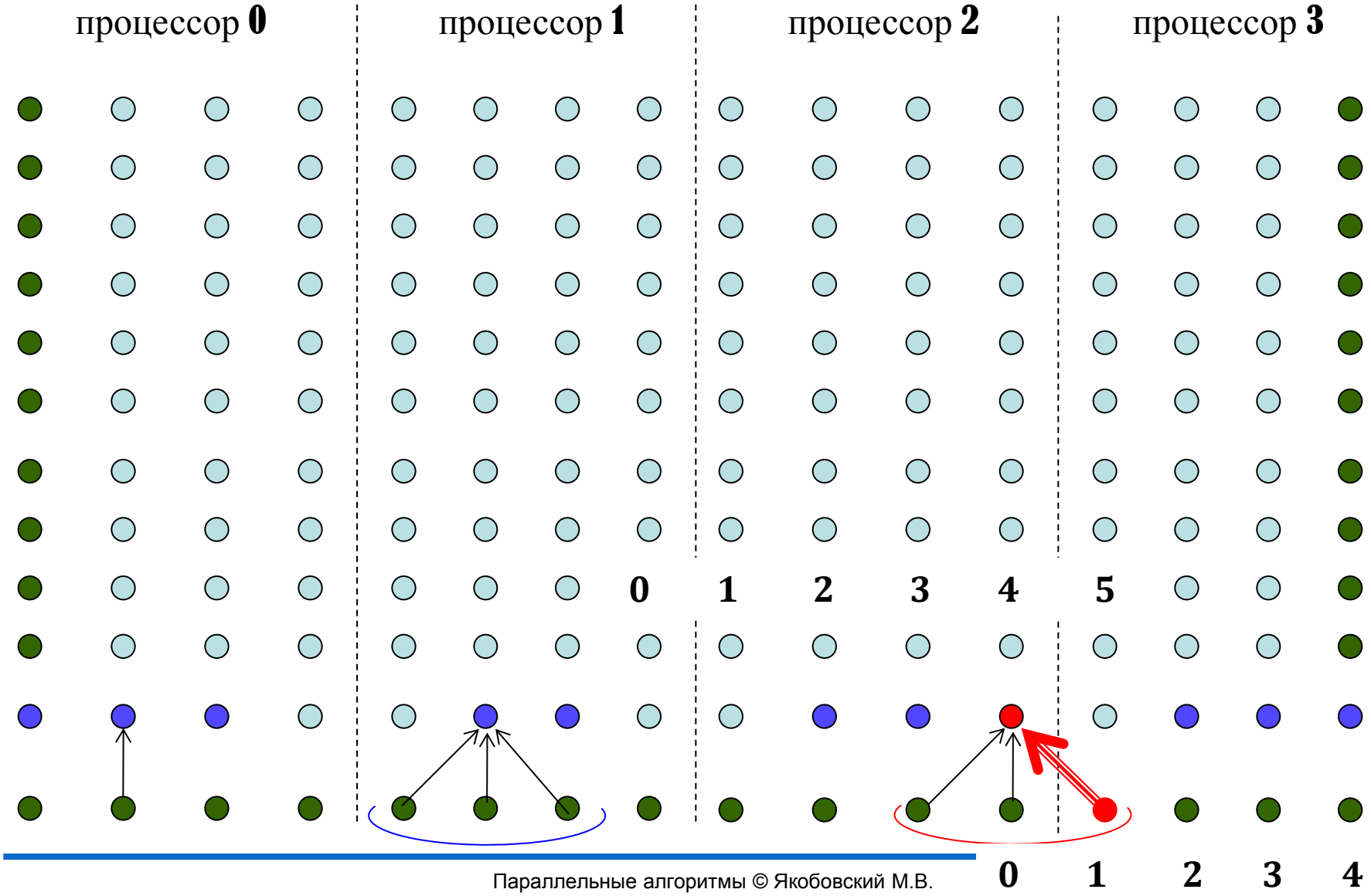
τ_c – время вычисления $fun(f-, f, f+)$



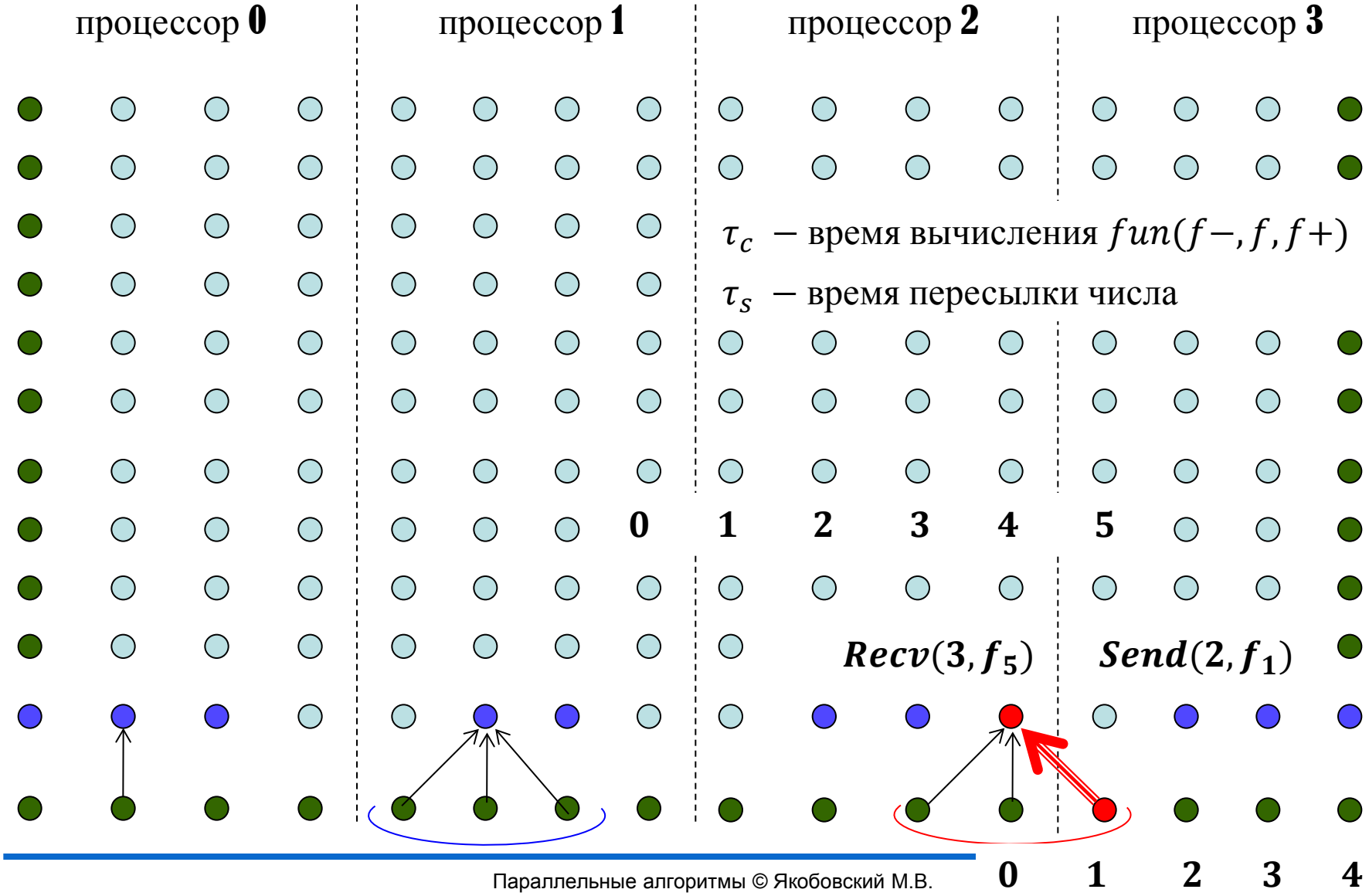
Метод геометрического параллелизма



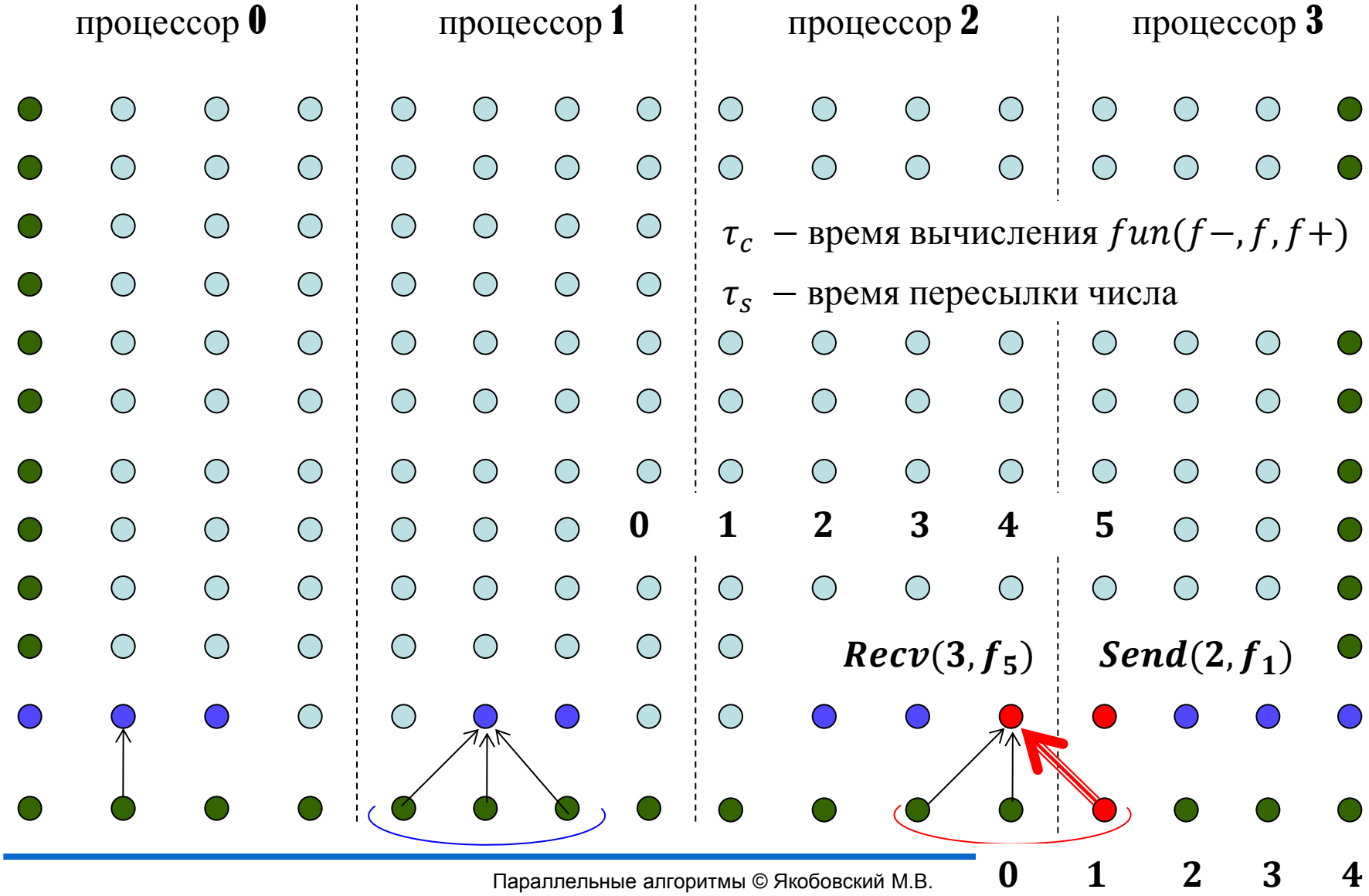
Метод геометрического параллелизма



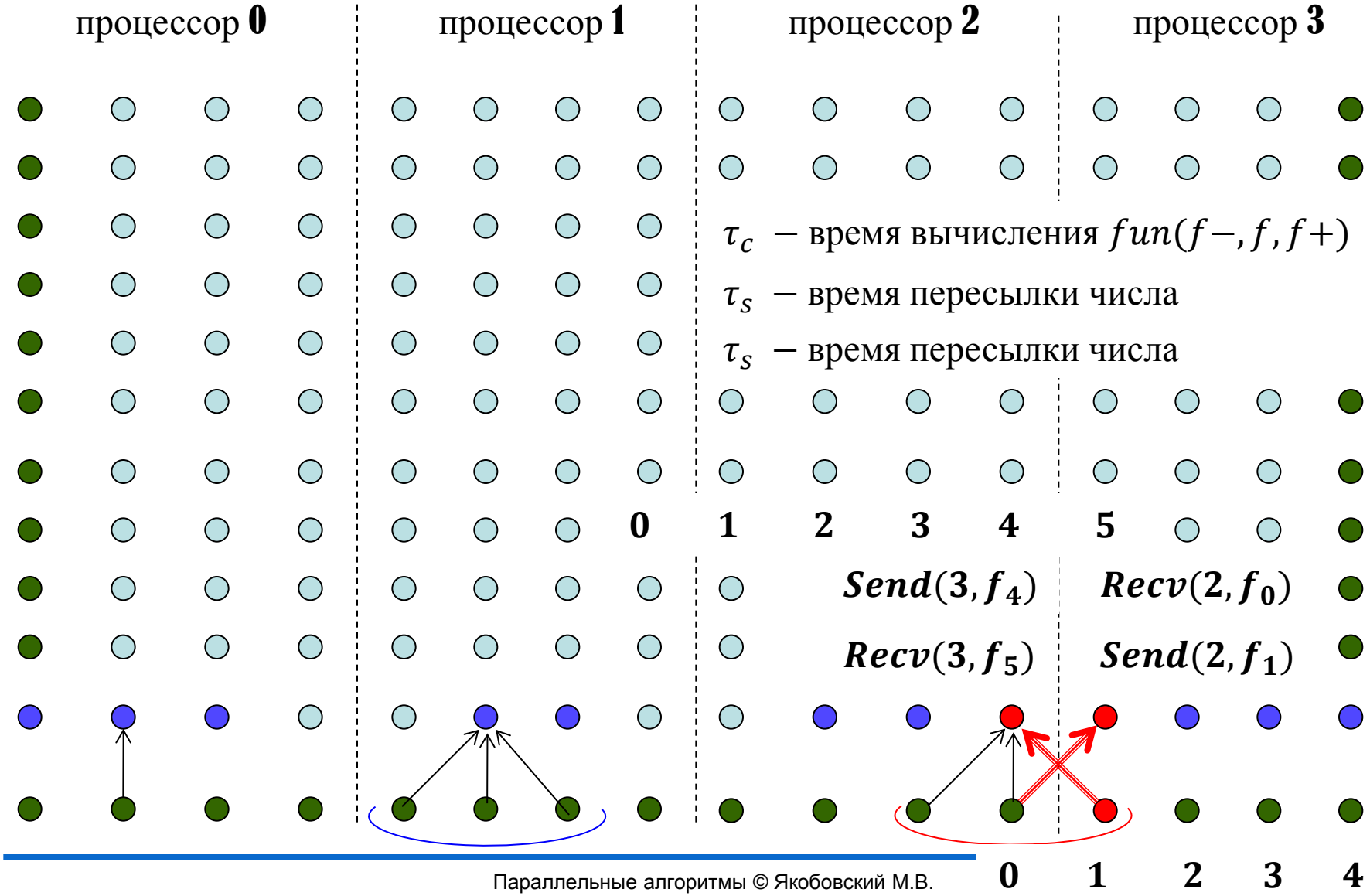
Метод геометрического параллелизма



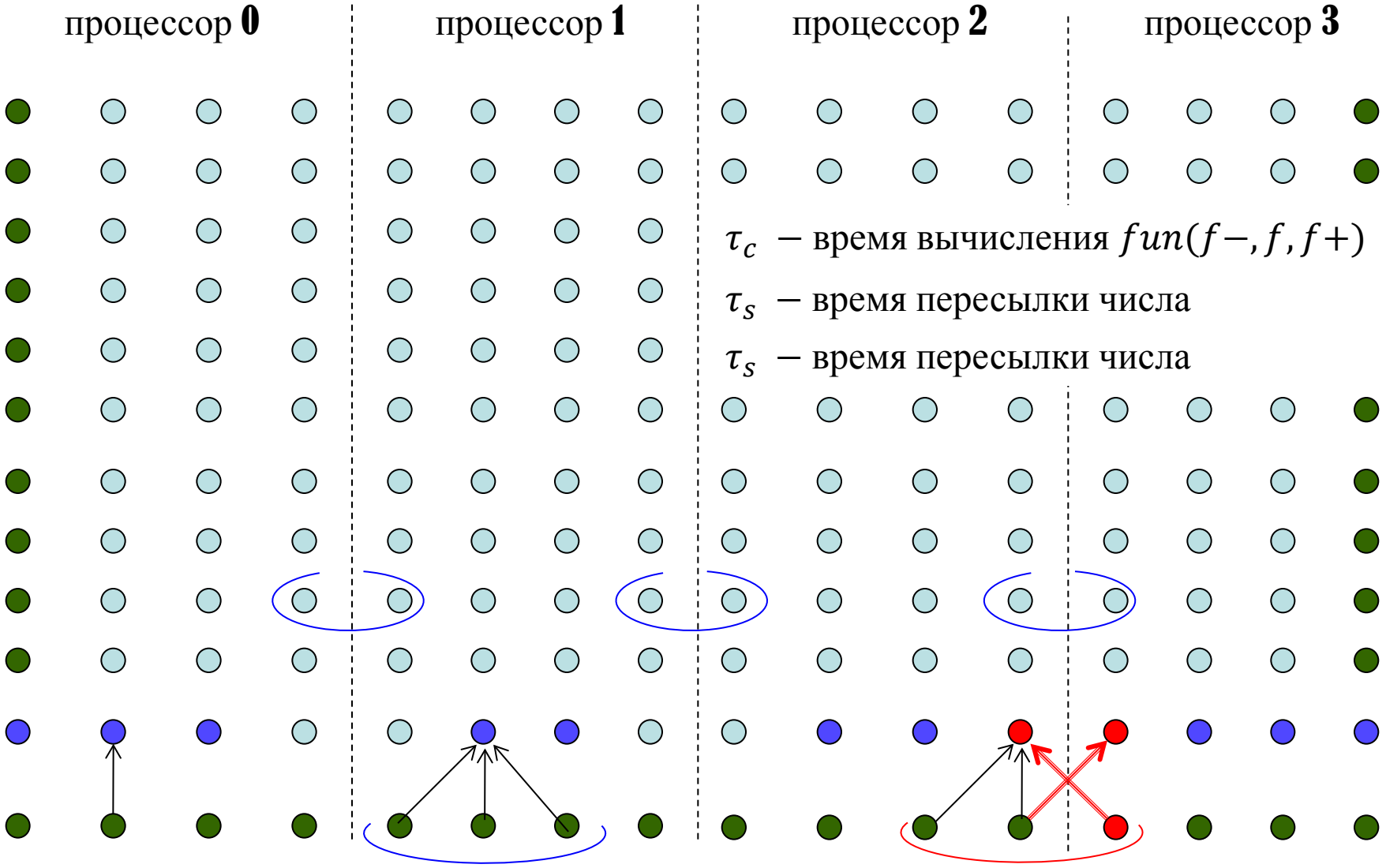
Метод геометрического параллелизма



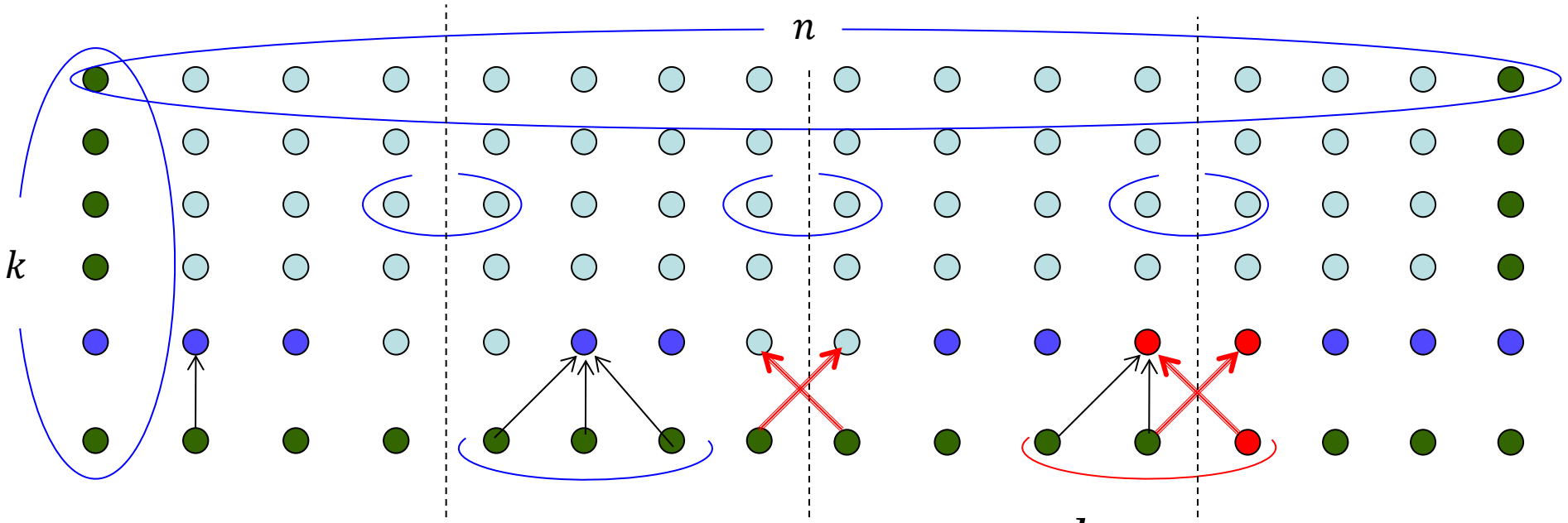
Метод геометрического параллелизма



Метод геометрического параллелизма



Метод геометрического параллелизма p – число процессов



$$T_1(kn) = \tau_c kn \qquad T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

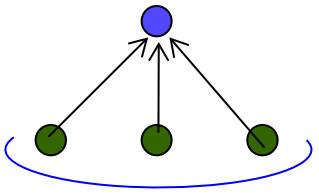
$$S_p(kn) = p\frac{1}{1+4\frac{p}{n}\frac{\tau_s}{\tau_c}} \qquad E_p(kn) = \frac{1}{1+4\frac{p}{n}\frac{\tau_s}{\tau_c}}$$

Верно или нет ?

$$T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

```
for( шаг = 0; шаг < k ; шаг++ )
{
   for( i=rank*n/p; i<(rank+1)*n/p ; i++)
     fun( i-1, i, i+1 )

   if(rank > 0   )  Send(rank-1, f )
   if(rank < p-1 )  Send(rank+1, f )
   if(rank > 0   )  Recv(rank-1, f )
   if(rank < p-1 )  Recv(rank+1, f )
}
```

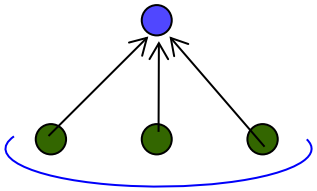


$$T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

Верно ли, что именно **4** ?

```
for(шаг=0;шаг<k;шаг++)
{
   for( i=rank*n/p; i<(rank+1)*n/p ; i++)
      fun( i-1, i, i+1 )

   if(rank > 0   )  Send(rank-1, f )
   if(rank < p-1 )  Send(rank+1, f )
   if(rank > 0   )  Recv(rank-1, f )
   if(rank < p-1 )  Recv(rank+1, f )
}
```



0	1	2	3	4	5
	⇐	⇐	⇐	⇐	⇐
⇒	⇒	⇒	⇒	⇒	
	>=	>=	>=	>=	>=
=<	=<	=<	=<	=<	

$$T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s \longleftarrow$$

Верно ли, что именно **4** ?

```
for(шаг=0;шаг<k;шаг++)
{
   for( i=rank*n/p; i<(rank+1)*n/p ; i++)
      fun( i-1, i, i+1 )

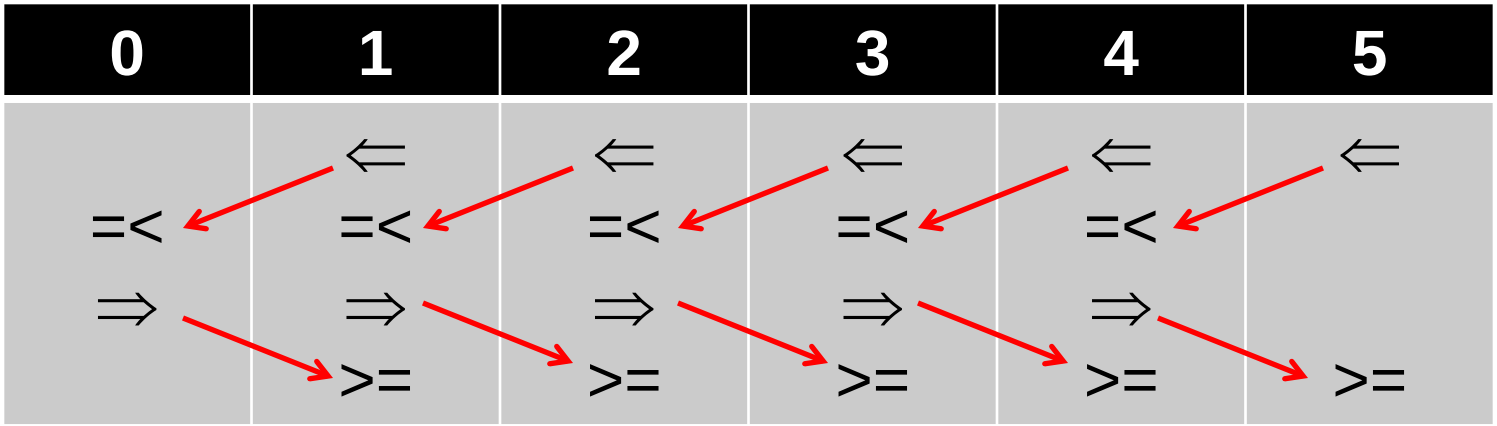
   if(rank > 0   )  Send(rank-1, f )
   if(rank < p-1 )  Send(rank+1, f )
   if(rank > 0   )  Recv(rank-1, f )
   if(rank < p-1 )  Recv(rank+1, f )
}
```

0	1	2	3	4	5
	⇐	⇐	⇐	⇐	⇐
⇒	⇒	⇒	⇒	⇒	
	>=	>=	>=	>=	>=
=<	=<	=<	=<	=<	

$$T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

```
for(шаг=0;шаг<k;шаг++)
{
    for(кирпич=rank*n/p;кирпич<(rank+1)*n/p;кирпич++)
        Уложить(кирпич)

    if(rank>0)     Send(rank-1, кирпич уложен! )
    if(rank<p-1)   Recv(rank+1, место готово?  )
    if(rank<p-1)   Send(rank+1, кирпич уложен! )
    if(rank>0)     Recv(rank-1, место готово?  )
}
```



$$T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

0	1	2	3	4	5
=< ⇒	⇐ =< ⇒ >=	⇐ =< ⇒ >=	⇐ =< ⇒ >=	⇐ =< ⇒ >=	⇐ >=

$$T_p(kn) = \tau_c \frac{kn}{p} + 1k\tau_s$$

0	1	2	3	4	5
=<	⇐				
⇒	=<	⇐	⇐	⇐	⇐
	⇒	=<	=<	=<	>=
	>=	⇒	⇒	⇒	
		>=	>=	>=	

$$T_p(kn) = \tau_c \frac{kn}{p} + 2k\tau_s$$

0	1	2	3	4	5
=<	⇐				
	=<	⇐			
⇒	⇒	=<	⇐	⇐	⇐
	>=	⇒	=<	=<	>=
		>=	⇒	⇒	
			>=	>=	

$$T_p(kn) = \tau_c \frac{kn}{p} + 3k\tau_s$$

0	1	2	3	4	5
=<	⇐				
	=<	⇐			
		=<	⇐		
⇒	⇒	⇒	=<	⇐	⇐
	>=	>=	⇒	=<	>=
			>=	⇒	
				>=	

$$T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

0	1	2	3	4	5
=<	⇐				
	=<	⇐			
		=<	⇐		
			=<	⇐	
⇒	⇒	⇒	⇒	=<	⇐
	>=	>=	>=	⇒	>=
				>=	

$$T_p(kn) = \tau_c \frac{kn}{p} + pk\tau_s$$

0	1	2	3	4	5
=<	⇐				
	=<	⇐			
		=<	⇐		
			=<	⇐	
				=<	⇐
⇒	⇒	⇒	⇒	⇒	>=
	>=	>=	>=	>=	

$$T_p(kn) = \tau_c \frac{kn}{p} + (p+1)k\tau_s$$

0	1	2	3	4	5
=<	⇐				
	=<	⇐			
		=<	⇐		
			=<	⇐	
				=<	⇐
				⇒	>=
⇒	⇒	⇒	⇒	>=	
	>=	>=	>=		

$$T_p(kn) = \tau_c \frac{kn}{p} + (p+2)k\tau_s$$

0	1	2	3	4	5
=<	⇐	⇐	⇐	⇐	⇐
	=<	=<	=<	=<	>=
⇒	⇒	⇒	⇒	⇒	
	>=	>=	>=	>=	

$$T_p(kn) = \tau_c \frac{kn}{p} + (p+3)k\tau_s$$

0	1	2	3	4	5
=<	⇐	⇐	⇐	⇐	⇐
	=<	=<	=<	=<	>=
				⇒	
⇒	⇒	⇒	⇒	>=	
	>=	>=	>=		

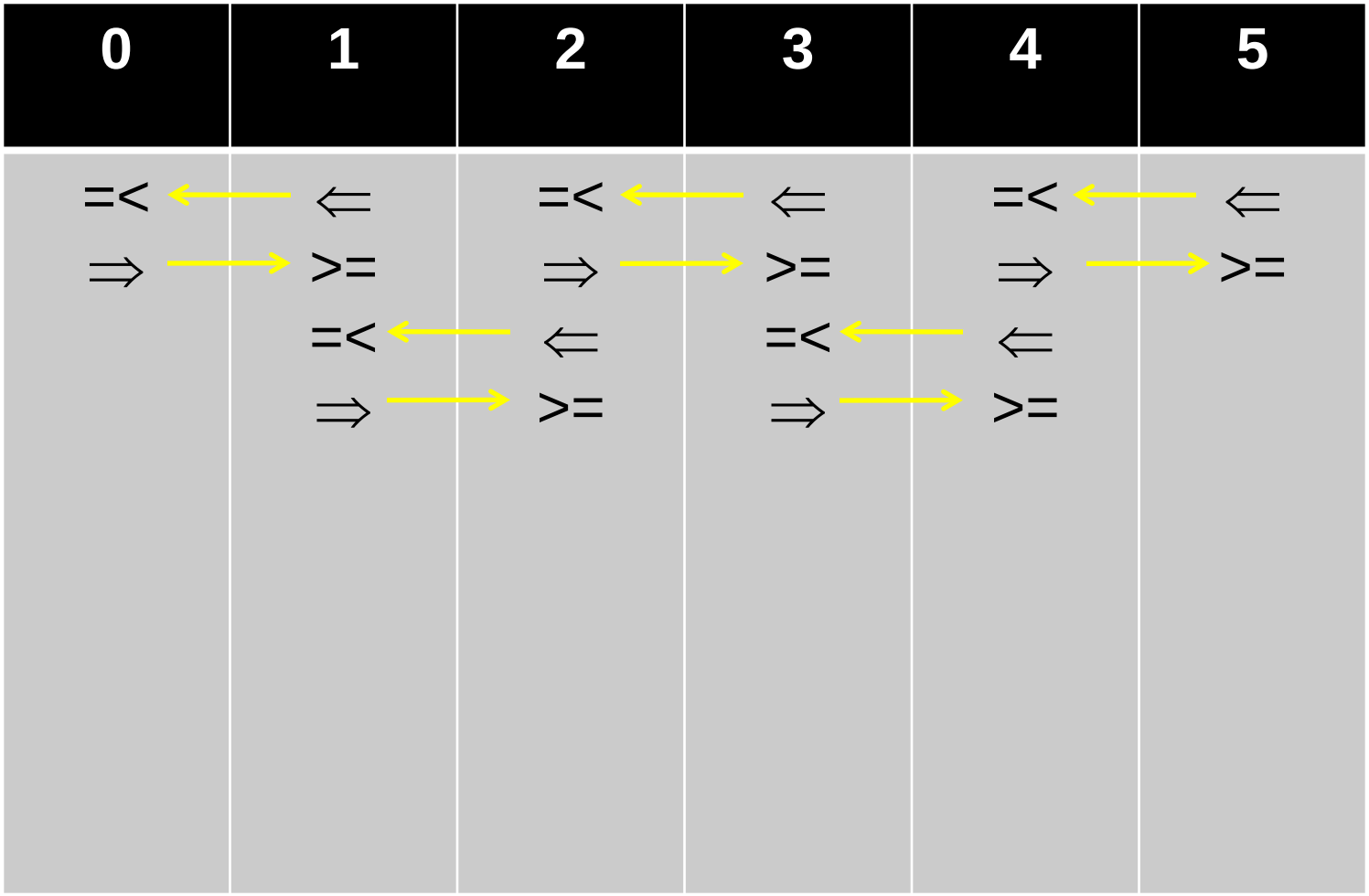
$$T_p(kn) = \tau_c \frac{kn}{p} + (p+4)k\tau_s$$

0	1	2	3	4	5
=<	⇐	⇐	⇐	⇐	⇐
	=<	=<	=<	=<	>=
				⇒	
	⇒	⇒	⇒	>=	
⇒	>=	>=	>=		

$$T_p(kn) = \tau_c \frac{kn}{p} + 2pk\tau_s$$

0	1	2	3	4	5
=<	⇐	⇐	⇐	⇐	⇐
	=<	=<	=<	=<	>=
				⇒	
			⇒	>=	
		⇒	>=		
	⇒	>=			
⇒	>=				

$$T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

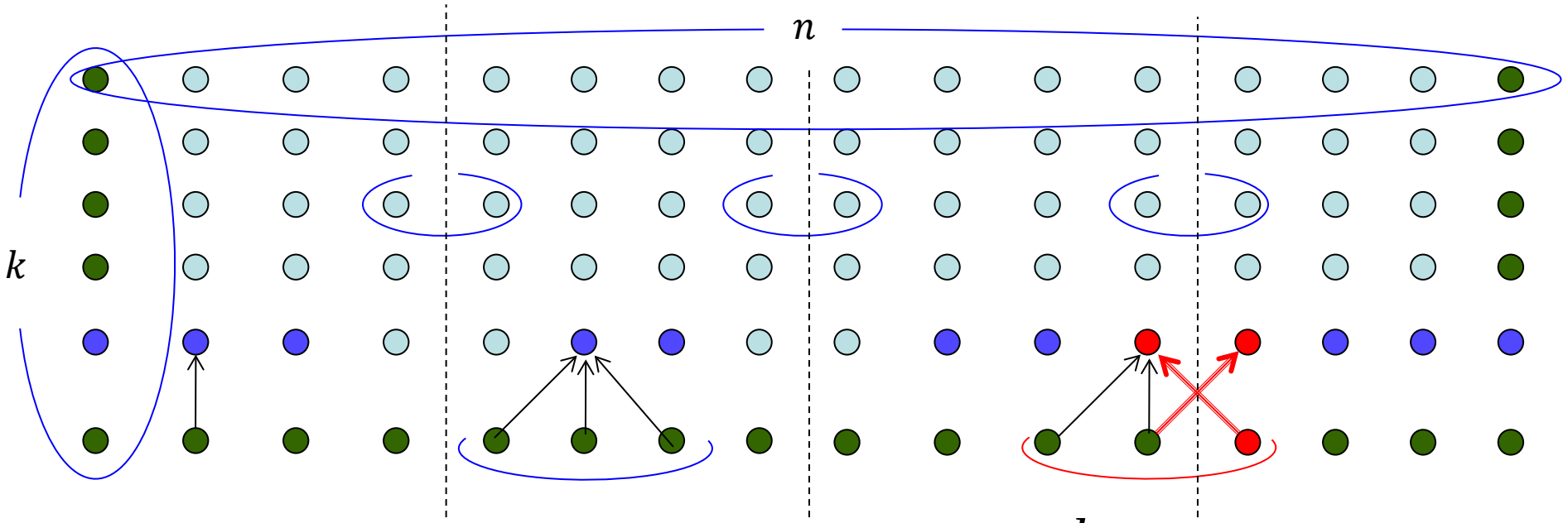


$$T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

```
for(шаг=0;шаг<k;шаг++)
{
    for(кирпич=rank*n/p;кирпич<(rank+1)*n/p;кирпич++)
       Уложить(кирпич)

    if(rank%2)
    {
    if(rank>0)    Send(rank-1, кирпич уложен! )
    if(rank>0)    Recv(rank-1, место готово?  )
    if(rank<p-1)  Recv(rank+1, место готово?  )
    if(rank<p-1)  Send(rank+1, кирпич уложен! )
    }
    else
    {
    if(rank<p-1)  Recv(rank+1, место готово?  )
    if(rank<p-1)  Send(rank+1, кирпич уложен! )
    if(rank>0)    Send(rank-1, кирпич уложен! )
    if(rank>0)    Recv(rank-1, место готово?  )
    }
}
```

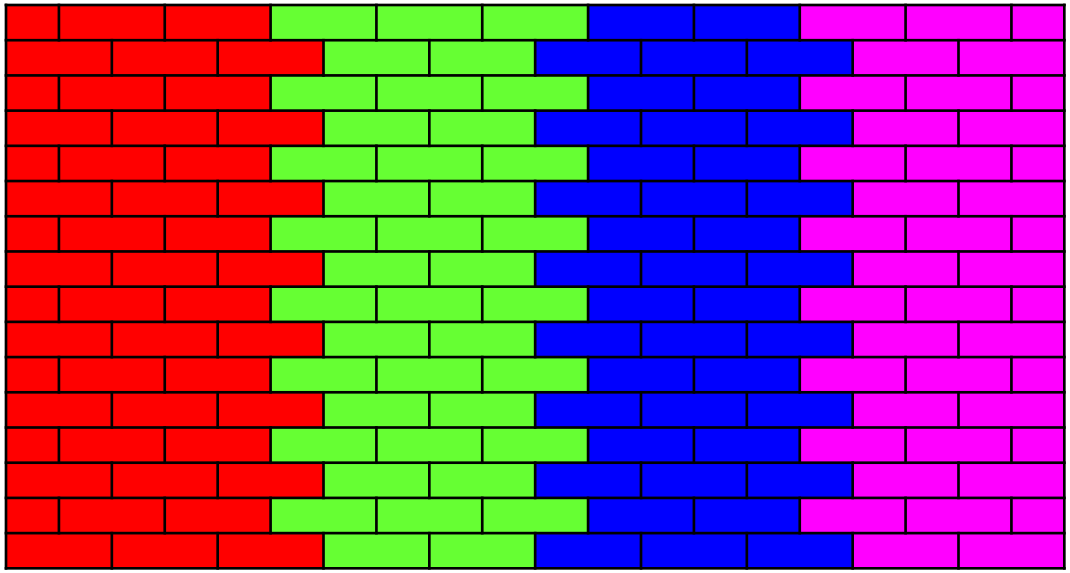
Метод геометрического параллелизма p – число процессов



$$T_1(kn) = \tau_c kn \qquad T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

$$S_p(kn) = p\frac{1}{1+4\frac{p}{n}\frac{\tau_s}{\tau_c}} \qquad E_p(kn) = \frac{1}{1+4\frac{p}{n}\frac{\tau_s}{\tau_c}}$$

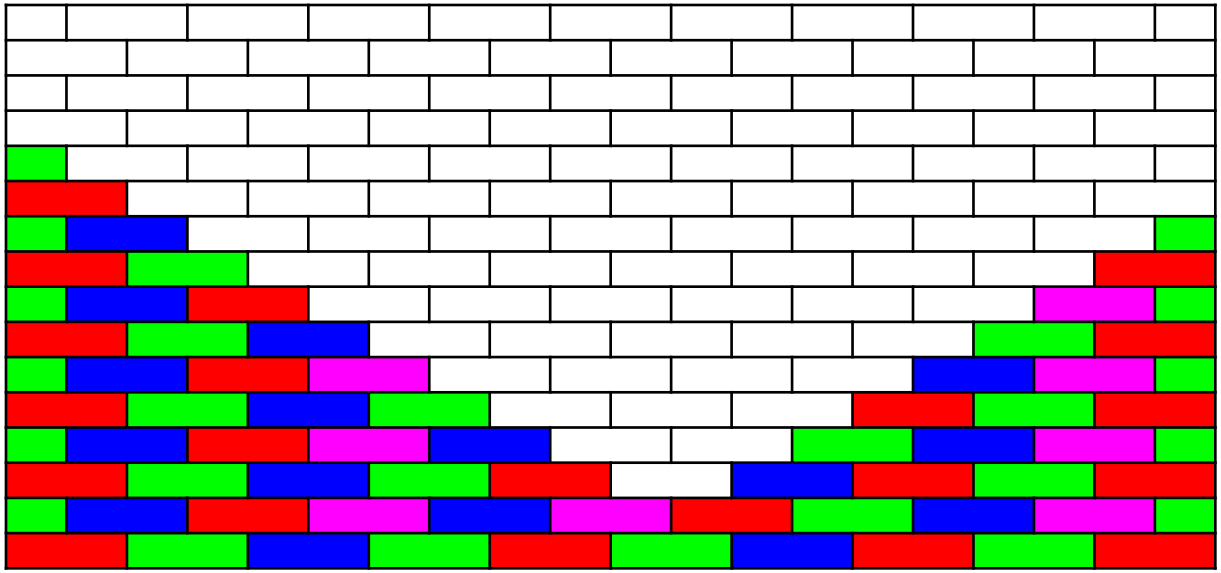
Возможные причины потери эффективности?



$$T_1(kn) = \tau_c kn \qquad T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

$$S_p(kn) = p \frac{1}{1 + 4\frac{p}{n}\frac{\tau_s}{\tau_c}} \qquad E_p(kn) = \frac{1}{1 + 4\frac{p}{n}\frac{\tau_s}{\tau_c}}$$

Возможные причины потери эффективности?



n – ширина стены

κ – высота стены

Контакты

Якобовский М.В., *чл.-корр. РАН, проф., д.ф.-м.н.,*

Заместитель директора по научной работе
Института прикладной математики
им. М.В.Келдыша Российской академии наук

mail: lira@imamod.ru

web: http://lira.imamod.ru