

**Интернет Университет  
Суперкомпьютерных технологий**

**Учебный курс**

***Введение в параллельные алгоритмы***

**Лекция 2**

**Методы построения параллельных программ**

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

# Предварительные замечания

---

*... 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*

# Содержание лекции

---

- Методы построения параллельных алгоритмов и их свойства:
  - Статическая балансировка
    - метод сдваивания
    - геометрический параллелизм
    - конвейерный параллелизм
  - Динамическая балансировка
    - коллективное решение
- Пример задачи, для параллельного решения которой необходимо создание качественно нового алгоритма

# Хороший параллельный алгоритм

больша

- ❑ Обладает запасом внутреннего параллелизма
  - Есть возможность одновременного выполнения операций
- ❑ Допускает возможность равномерного распределения вычислительных операций между процессорами
- ❑ Обладает низким уровнем накладных расходов

большим

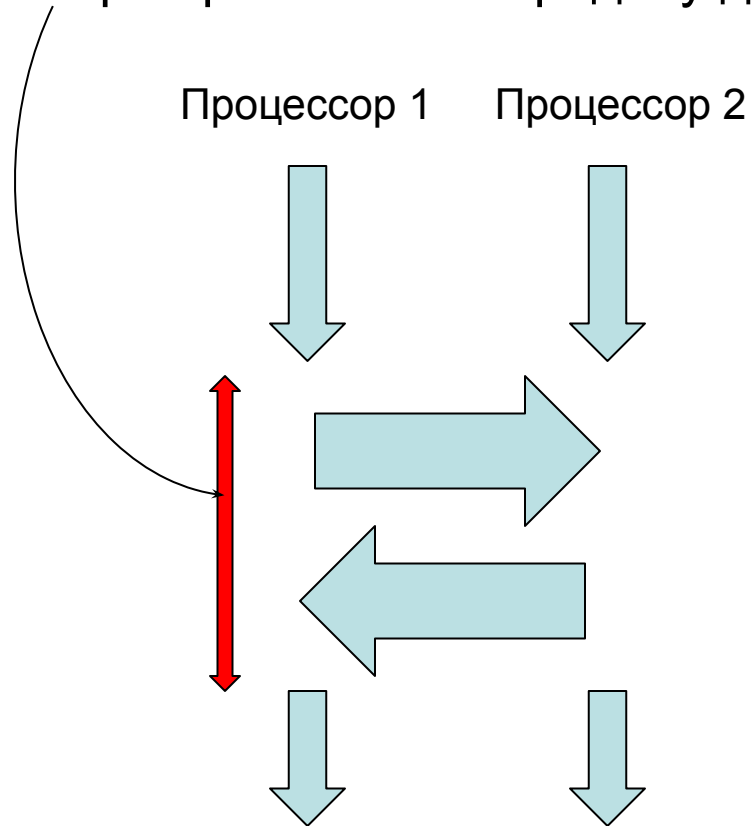
# Накладные расходы

---

- ❑ Операции, отсутствующие в наилучшем последовательном алгоритме:
  - Синхронизация
  - Обмен данными
  - Дублирование операций
  - Новые операции

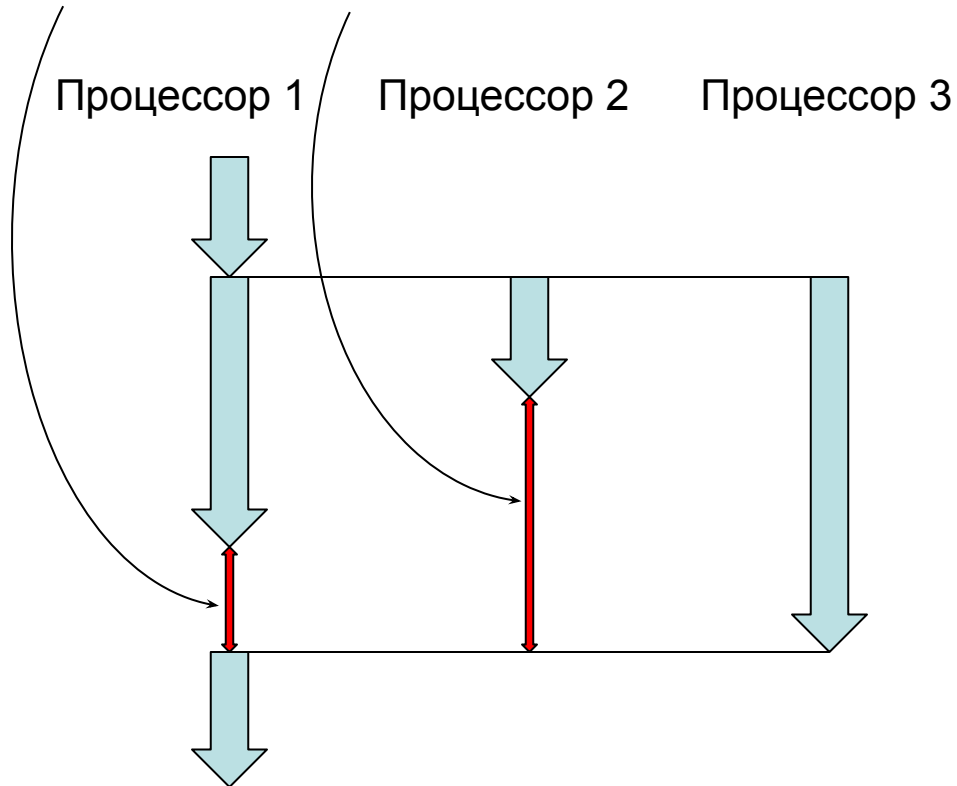
# Обмен данными

- Потери времени на передачу данных между процессами



# Синхронизация

- ❑ Потери времени на ожидание долго выполняющихся процессов



# Дублирование операций

**S=0;**

For(i=0;i<n1;i++)

S+=a[i];

Send(S)

**S=0;**

For(i=n1;i<n;i++)

S+=a[i];

Send(S)

Recv(S1)

Recv(S2)

S=S1+S2



# Вычисление всех факториалов до 8! включительно

Шаг	Процессор 1	Процессор 2	Процессор 3	Процессор 4
1	1 · 2	3 · 4	5 · 6	7 · 8
2	12 · 3	12 · 34	56 · 7	56 · 78
3	1234 · 5	1234 · 56	1234 · 567	1234 · 567 8

```
F=1;
for(i=2; i <= n; i++)
    F*=i;
```

$$T_{p=n/2}(n) = \tau_c \log_2 n$$

$$S = \frac{n-1}{\log_2 n} \Big|_{\substack{n=8 \\ p=4}} = \frac{7}{3} < 4 = p$$

$$T_1(n) = \tau_c (n-1)$$

$$E_{p=4}(n=8) = \frac{7}{12}$$

# Вычисление всех факториалов до 8! включительно

Шаг	Процессор 1	Процессор 2	Процессор 3	Процессор 4
1	1 · 2	3 · 4	5 · 6	7 · 8
2	12 · 3	12 · 34	56 · 7	56 · 78
3	1234 · 5	1234 · 56	1234 · 567	1234 · 567

$T_{p=n/2}(n) = t_c \log_2 n$    
 $S = \frac{n-1}{\log_2 n} \Big|_{\substack{n=8 \\ p=4}} = \frac{7}{3} < 4 = p$    
 $E_{p=4}(n=8) = \frac{8}{12}$

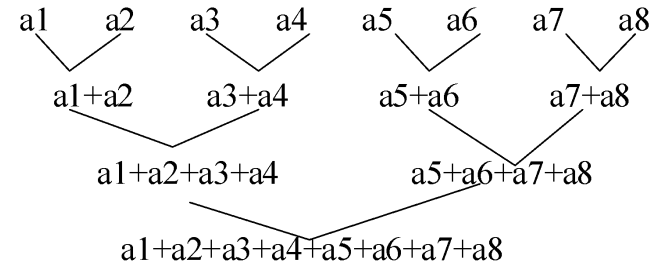
Шаг	Процессор 1	Процессор 2	Процессор 3	Процессор 4
1	1 2!	8 3 · 4	9 5 · 6	10 7 · 8
2	2 3!	3 4!	1 56 · 7	2 56 · 78
3	4 5!	5 6!	6 7!	7 8!

# Метод сдвигивания

## Каскадная схема

$$T_{p=n/2}(n) = \tau_c \log_2 n$$

$$S_{p=n/2}(n) = \frac{(n-1)}{\log_2 n} \quad E_{p=n/2}(n) \approx \frac{1}{\log_2 n}$$

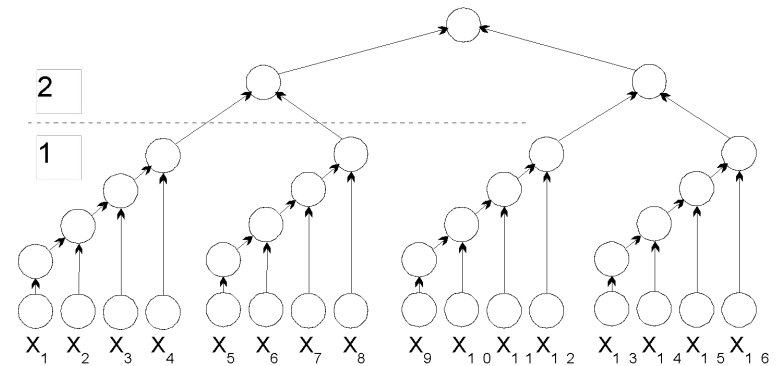


## Модифицированная каскадная схема

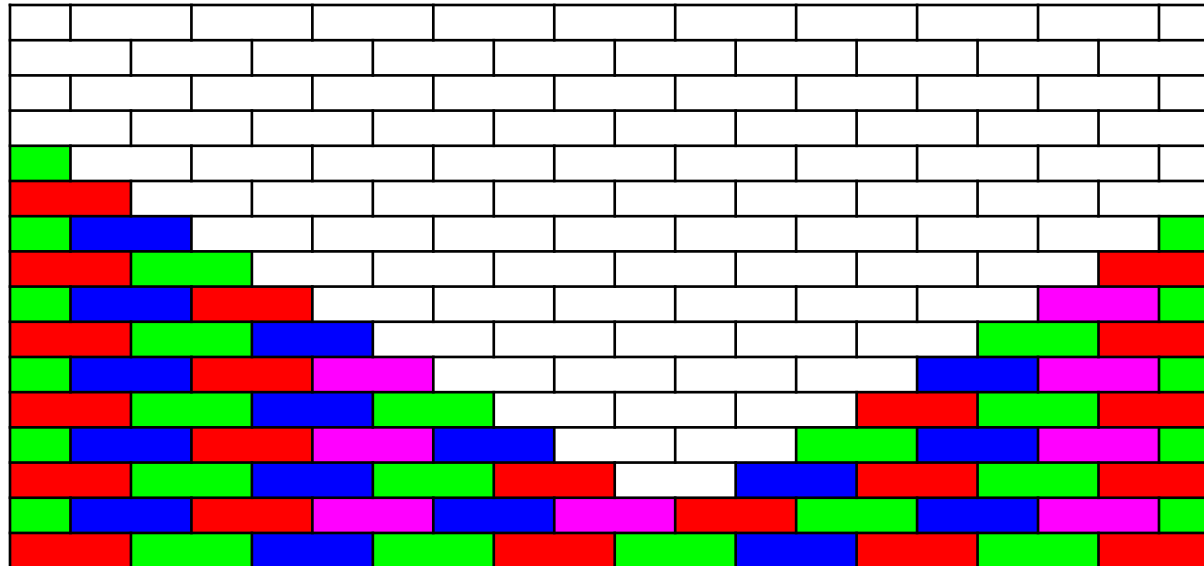
[В.П.Гергель Основы параллельных вычислений, лекция 4, слайд 23](#)

$$T_{p=\frac{n}{\log_2 n}}(n) \approx 2\tau_c \log_2 n$$

$$S_{p=\frac{n}{\log_2 n}}(n) \approx \frac{(n-1)}{2\log_2 n} \quad E_{p=\frac{n}{\log_2 n}}(n) \approx \frac{1}{2}$$



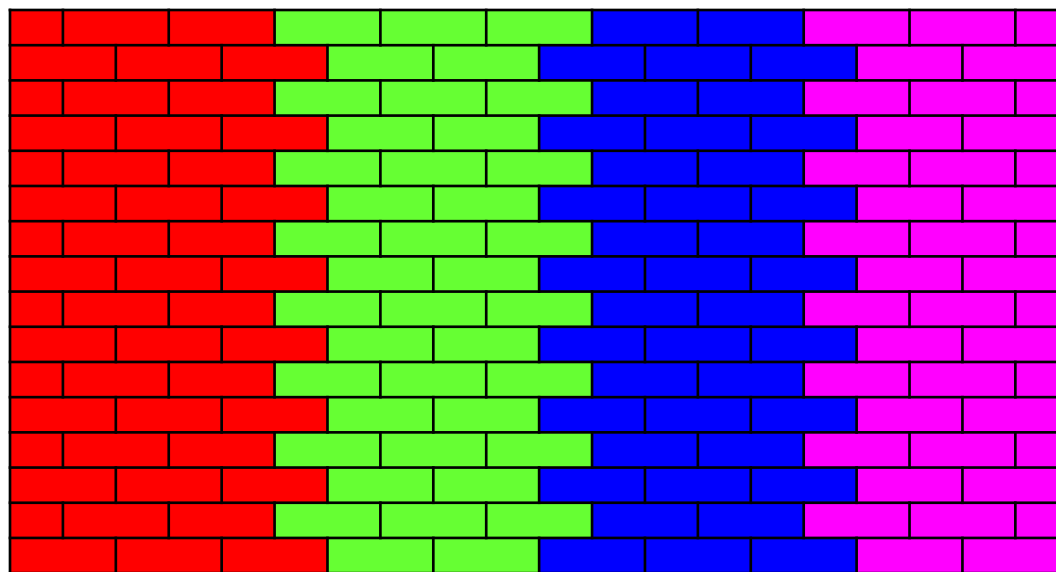
# Стена Фокса



$n$  – ширина стены

$k$  – высота стены

# Метод геометрического параллелизма



$$T_1(kn) = \tau_c kn$$

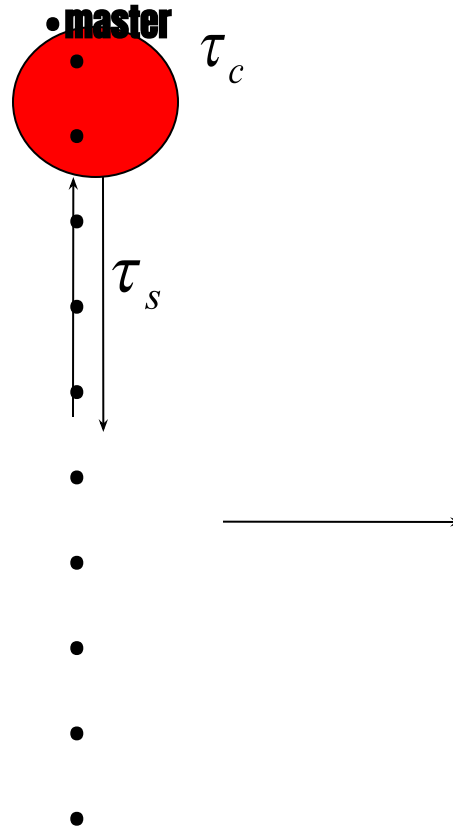
$$T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_s$$

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

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

# Метод коллективного решения (укладка паркета)

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



$$p_{\max} = \frac{\tau_c}{\tau_s}$$

# Метод коллективного решения (укладка паркета)

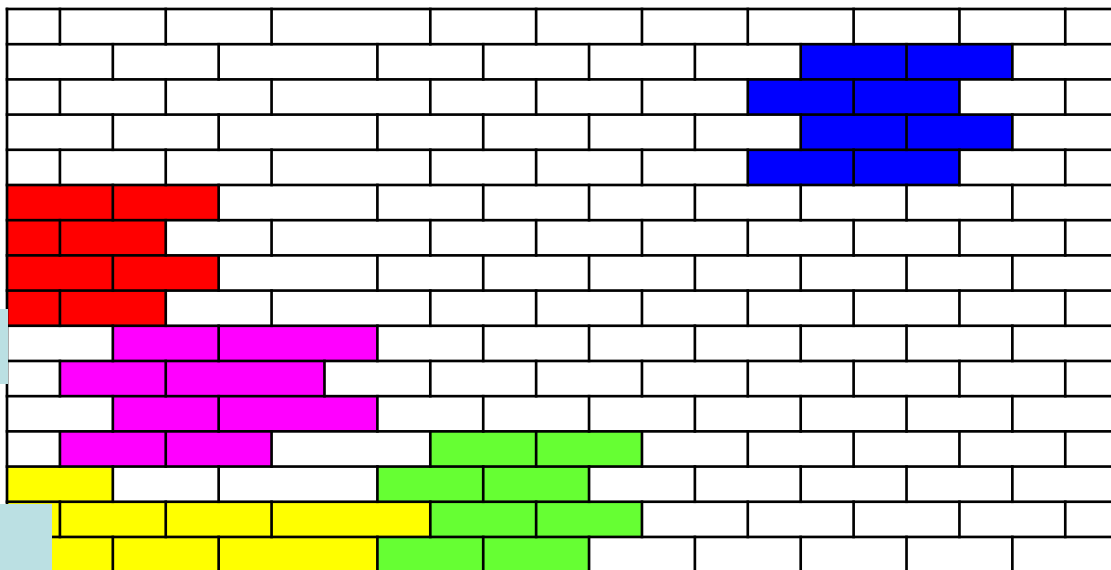
$$T_1(kn) = \tau_c kn$$

$$T_p(kn) = \frac{kn}{rp} (r\tau_c + \tau_s)$$

Число порций

Обработка порции

Обмен данными



$$S_{p=\frac{r\tau_c}{\tau_s}}(kn) = \left( \frac{r\tau_c}{\tau_s} \right)^2 \frac{1}{1 + \frac{r\tau_c}{\tau_s}} = p_{\max} \frac{1}{1 + \frac{1}{p_{\max}}}$$

$$p_{\max} = \frac{r\tau_c}{\tau_s}$$

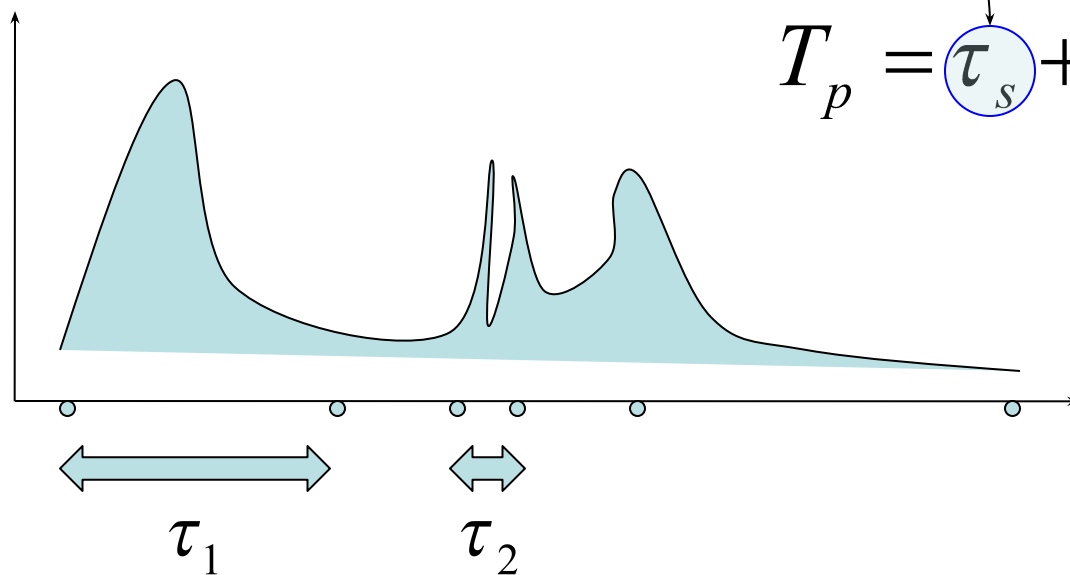
$$E_{p=\frac{r\tau_c}{\tau_s}}(kn) = \frac{1}{1 + \frac{\tau_s}{r\tau_c}}$$

$r$  - размер порции

# Вычисление определенного интеграла

□ `Send( $a_i$ ); Send( $a_{i+1}$ ); Recv( $s$ );`

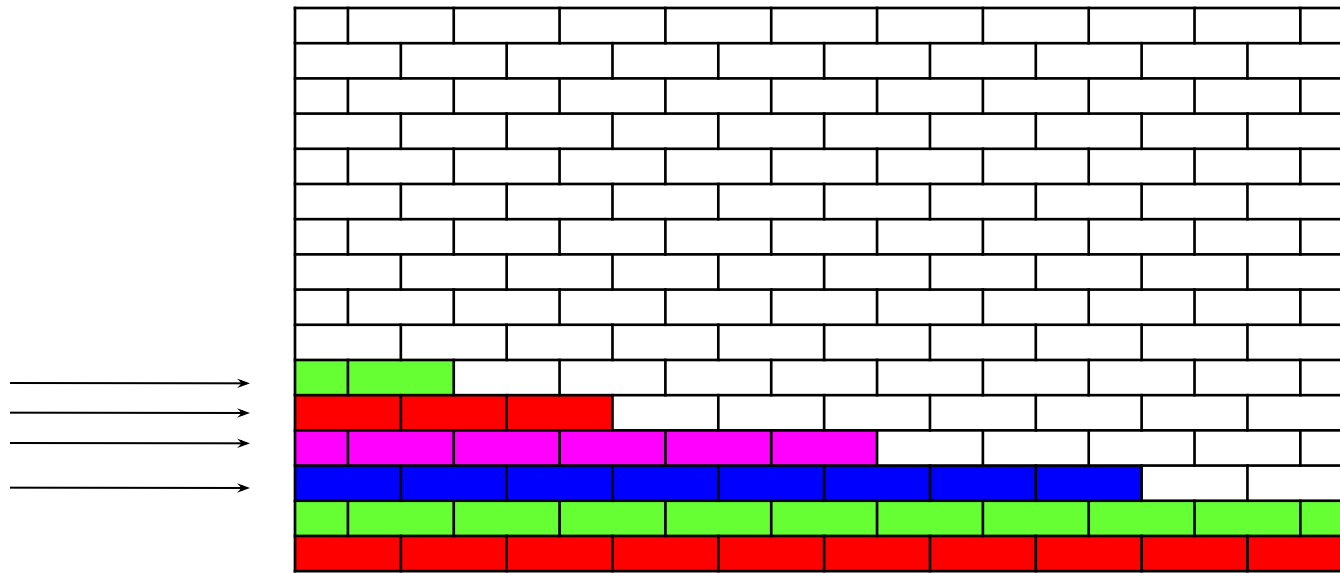
$$I = \int_A^B f(x) dx = \sum_i \int_{a_i}^{a_{i+1}} f(x) dx$$



$$T_p = \tau_s + \max_i \tau_i$$



# Метод конвейерного параллелизма



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

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

- Статическая и динамическая балансировка загрузки процессоров
  - Статическая балансировка
    - метод сдваивания
    - геометрический параллелизм
    - конвейерный параллелизм
  - Динамическая балансировка
    - коллективное решение

# Определение суммы двух многозначных чисел

```
r=0;  
for (i=0; i<=n; i++)  
  {  
    d=a[i]+b[i]+r;  
    c[i]=d%10;  
    r=d/10;  
  }  
c[i]=r;
```

```
+ 6934317835  
  3221643577  
-----  
10155961412
```

$$T_1 = 4n\tau_c$$

# «Параллельный» алгоритм

- Последовательное распространение разряда переноса на четырёх процессорах

$$\begin{array}{r} + 999999999 \\ \phantom{+} 1 \\ \hline 1000000000 \end{array}$$

99	99	99	99
			1
			100
		100	
	100		
100			
100	00	00	00

# Спекулятивный алгоритм

- Спекулятивное вычисление двух сумм

$$\begin{array}{r} + 999999999 \\ \phantom{+} \phantom{999999999} 1 \\ \hline 1000000000 \end{array}$$

99	99	99	99	
			1	
99	99	99	100	+0
100	100	100		+1
100	00	00	00	

# Спекулятивный алгоритм

```
r1=0;
r2=1;
for (i=0; i<=n1; i++)
  {
    d1=a[i]+b[i]+r1;
    c1[i]=d1%10;
    r1=d1/10;
    d2=a[i]+b[i]+r2;
    c2[i]=d2%10;
    r2=d2/10;
  }
Recv(&r)
if (r) c=c1;
else c=c2;
```

$$T' = \delta n_1 \tau_c$$

# Спекулятивный алгоритм

## □ Спекулятивное вычисление двух сумм

$$T_1 = 4n\tau_c$$

$$T_p = 8 \frac{n}{p} \tau_c$$

$$S_p = \frac{p}{2}$$

$$E_p = 50\%$$

99	99	99	99	
			1	
99	99	99	100	+0
100	100	100		+1
100	00	00	00	

# Заключение

---

- ❑ Рассмотрены методы построения параллельных алгоритмов
- ❑ Рассмотрена проблема балансировки загрузки процессоров
- ❑ Представлен масштабируемый параллельный метод сложения многоразрядных чисел, основанный на неэффективном последовательном алгоритме



# Вопросы для обсуждения

---

- ❑ В чем заключается проблема балансировки загрузки?
- ❑ В чем заключаются методы геометрического параллелизма, конвейерного параллелизма и коллективного решения?
- ❑ Чем определяются максимальные ускорения, достигаемые при применении этих методов?
- ❑ В чем отличие методов статической и динамической балансировки загрузки?

**Якобовский М.В.**

д.ф.-м.н.,

зав. сектором

«Программного обеспечения многопроцессорных систем и вычислительных сетей»

Института математического моделирования

Российской академии наук

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

[web: http://lira.imamod.ru](http://lira.imamod.ru)