

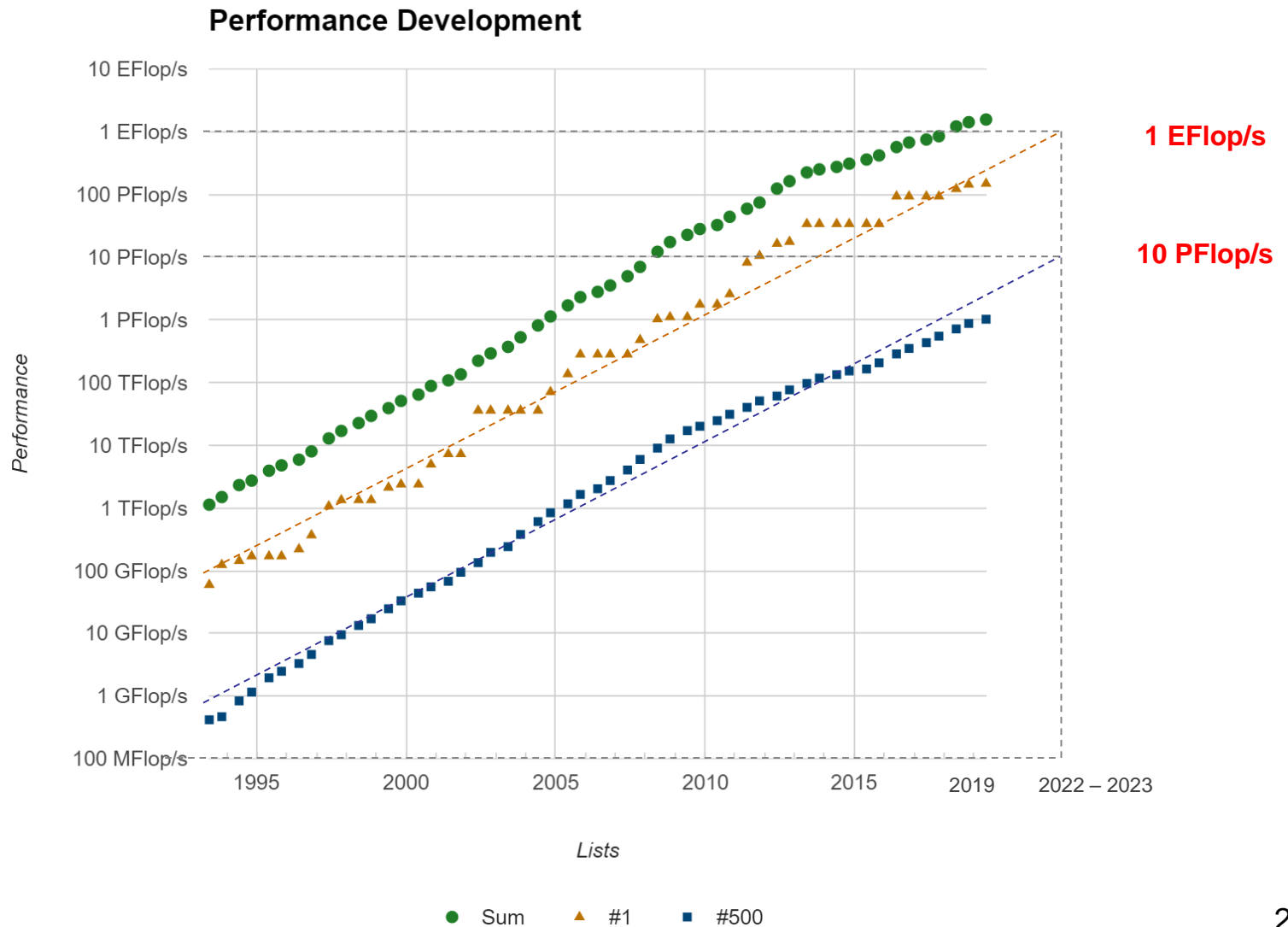


# BSF: модель параллельных вычислений для многопроцессорных систем с распределенной памятью

д.ф.-м.н., профессор Л.Б. Соколинский

Южно-Уральский государственный университет  
(национальный исследовательский университет)  
Челябинск

# TOP500 (июнь 2019)

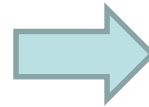


# Изменение основной парадигмы дизайна численных алгоритмов

---

Разработать алгоритм,  
эффективно работающий  
на малых вычислительных  
ресурсах

1985



Разработать алгоритм,  
способный эффективно  
использовать большие  
вычислительные ресурсы

2015

# Главная характеристика современного параллельного численного алгоритма

---

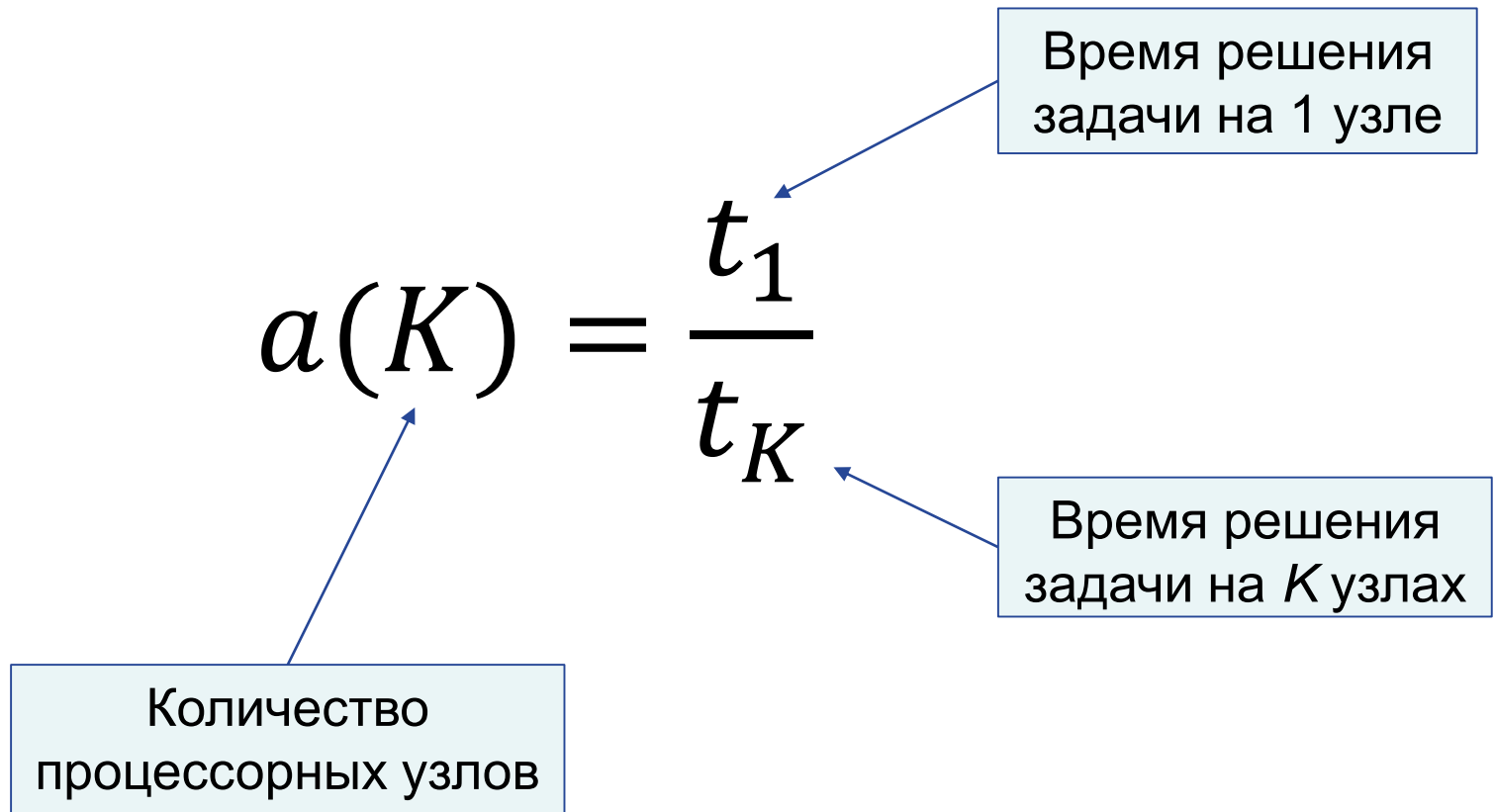
Масштабируемость



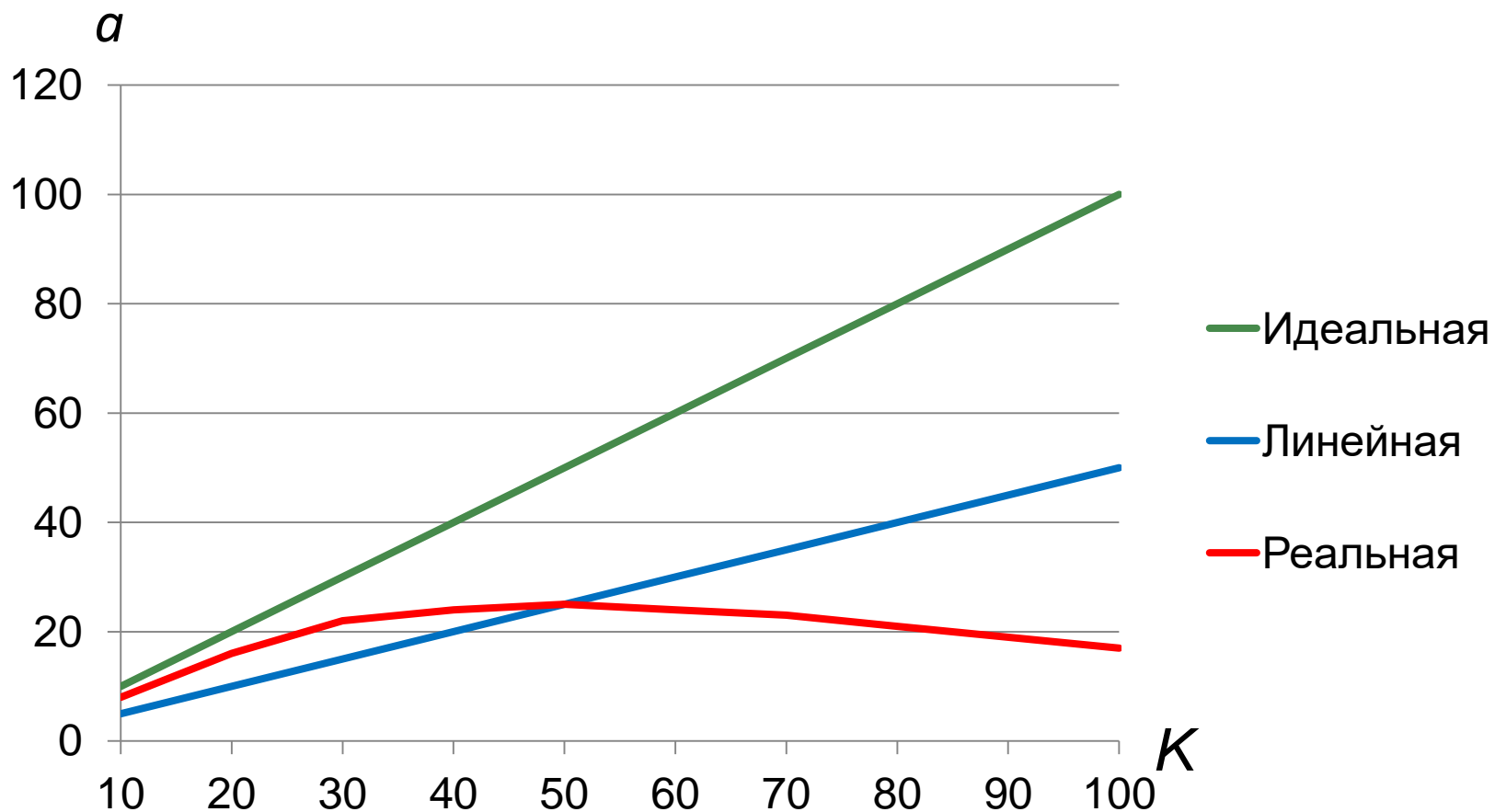
Кривая ускорения

# Ускорение

---



# Кривая ускорения – основной индикатор масштабируемости

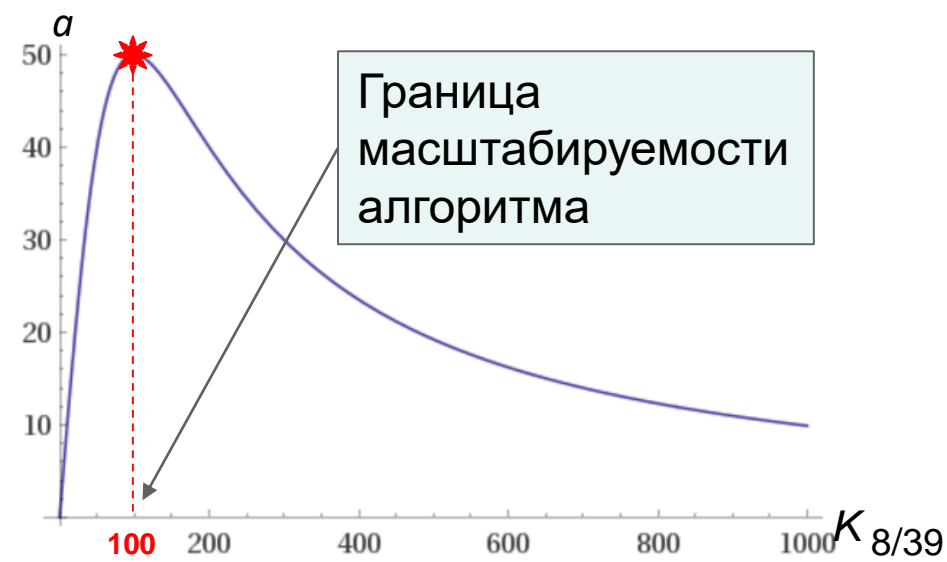
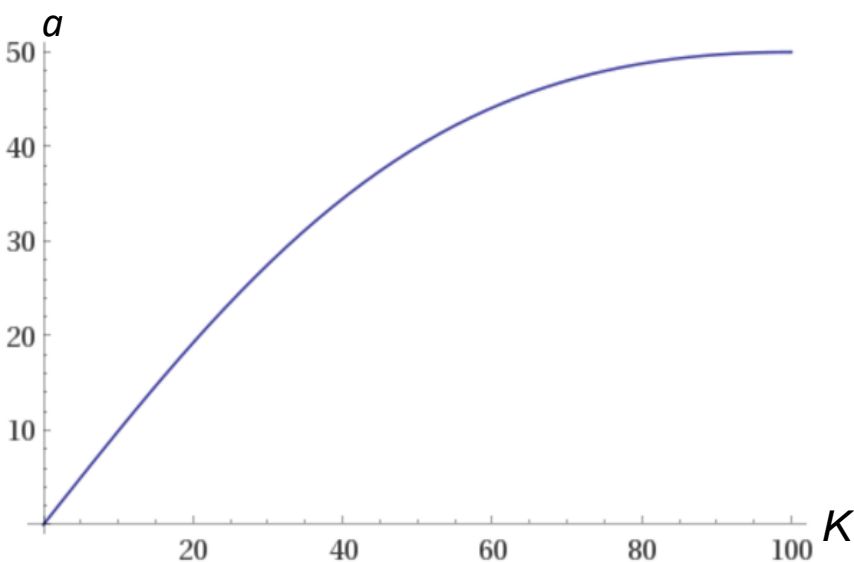
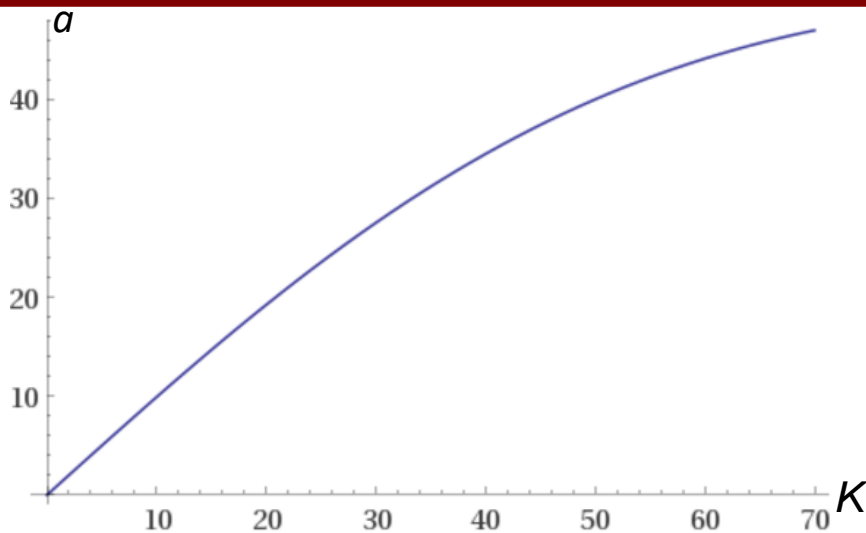
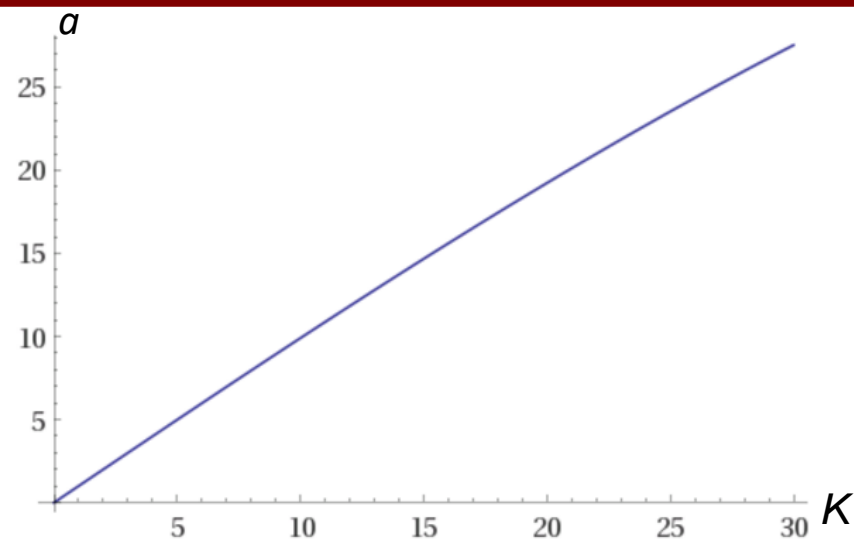


Алгоритм является хорошо масштабируемым, если:

---

кривая ускорения близка  
к линейной

# Кривая ускорения реальной задачи на кластерной вычислительной системе

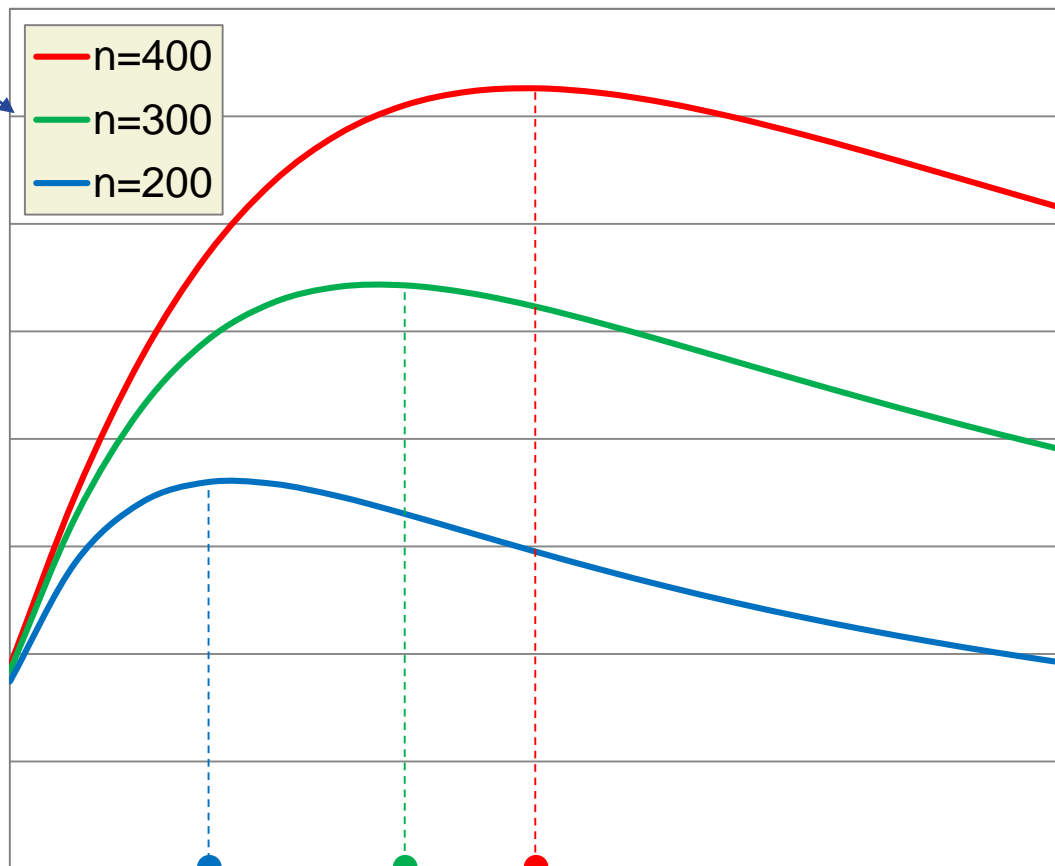




# Необходима функция $f$ для определения границы масштабируемости алгоритма **ДО** написания программы

$n$  - размер задачи

$a$



$$K_{max} = f(n)$$

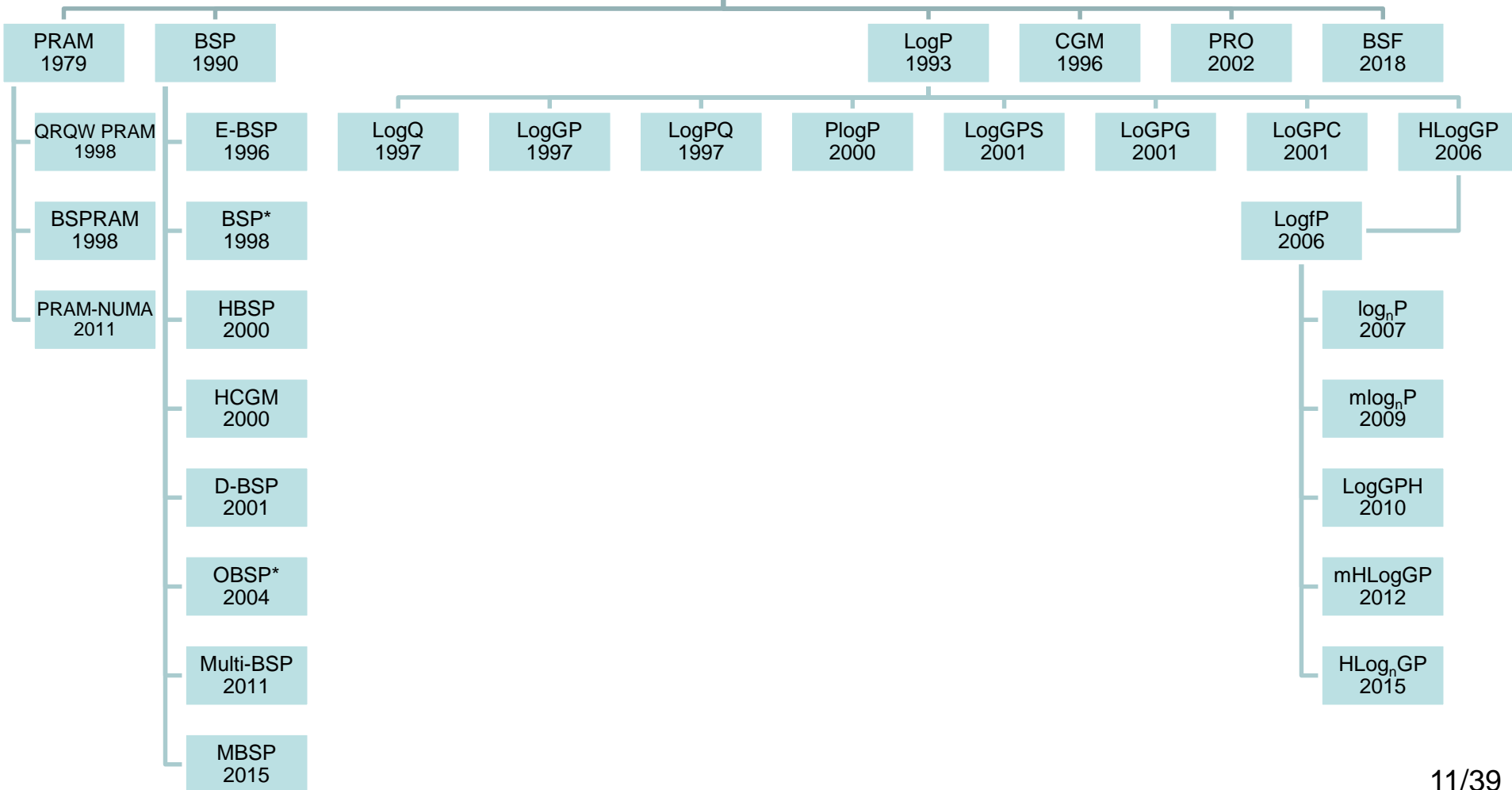
# Модель параллельных вычислений

---

- Модель параллельных вычислений – фреймворк (система правил и ограничений), включающий в себя:
  - абстрактное описание компьютера
  - правила написания алгоритмов и программ
  - способ их параллельного выполнения
  - стоимостную метрику для оценки времени выполнения
- Используется для разработки эффективных алгоритмов и программ
- Позволяет оценить реальную вычислительную сложность алгоритма и прогнозируемое быстродействие программы на различных вычислительных системах

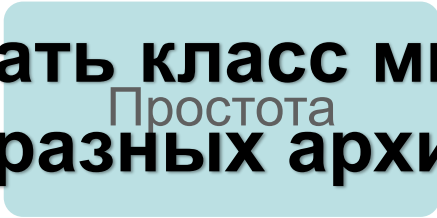

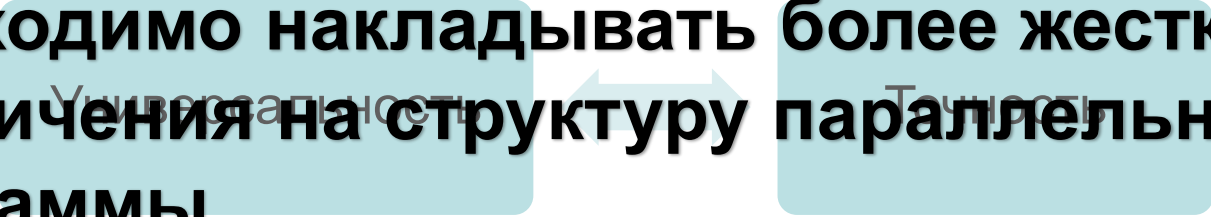
# Дерево моделей параллельных вычислений

## Модели параллельных вычислений



# Борьба противоположностей в моделях параллельных вычислений

---

1. Необходимо сужать класс многопроцессорных архитектур: для разных архитектур нужны разные модели  

2. Необходимо сужать класс алгоритмов: для разных алгоритмов нужны разные модели  

3. Необходимо накладывать более жесткие ограничения на структуру параллельной программы  


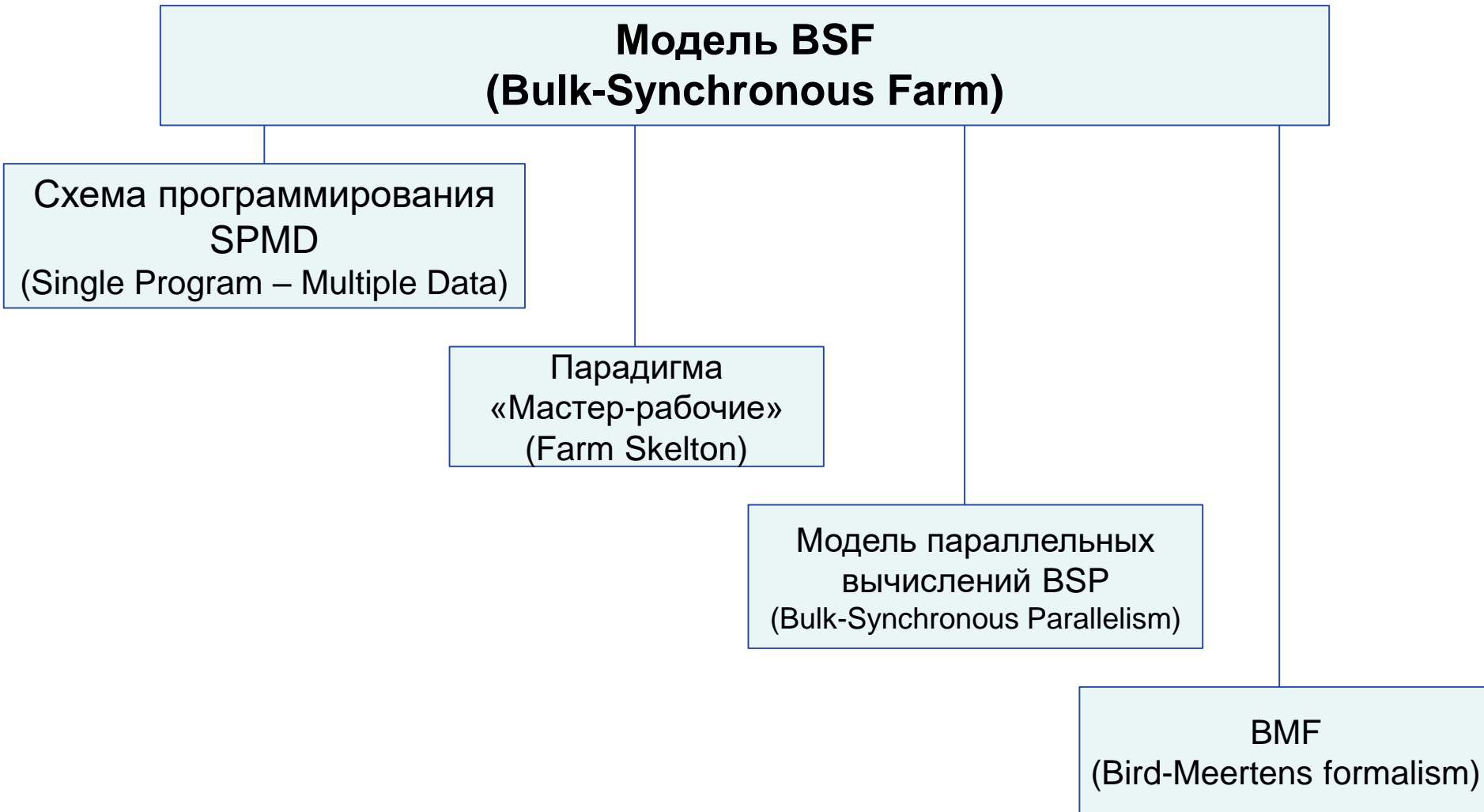
# Модель параллельных вычислений BSF (Bulk-Synchronous Farm)

---

- Область применения:
  - Многопроцессорные системы с распределенной памятью
  - Параллельные итерационные алгоритмы с высокой вычислительной сложностью
- Позволяет предсказать:
  - ускорение параллельного алгоритма
  - **границу масштабируемости параллельного алгоритма** (уникальное качество модели BSF)

# Фундамент модели BSF

---



# BSF-компьютер

---



Процессорные узлы

# Представление алгоритма в виде операций над списками

---

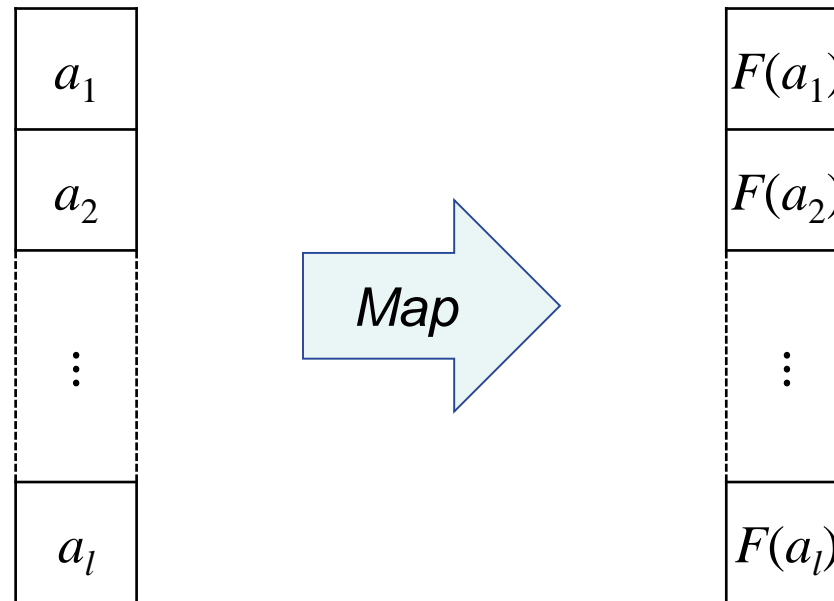
- Функции высшего порядка:
    - Map
    - Reduce
- (Формализм Бёрда-Миртенса)



# Функция высшего порядка *Map*

---

## Неограниченный параллелизм



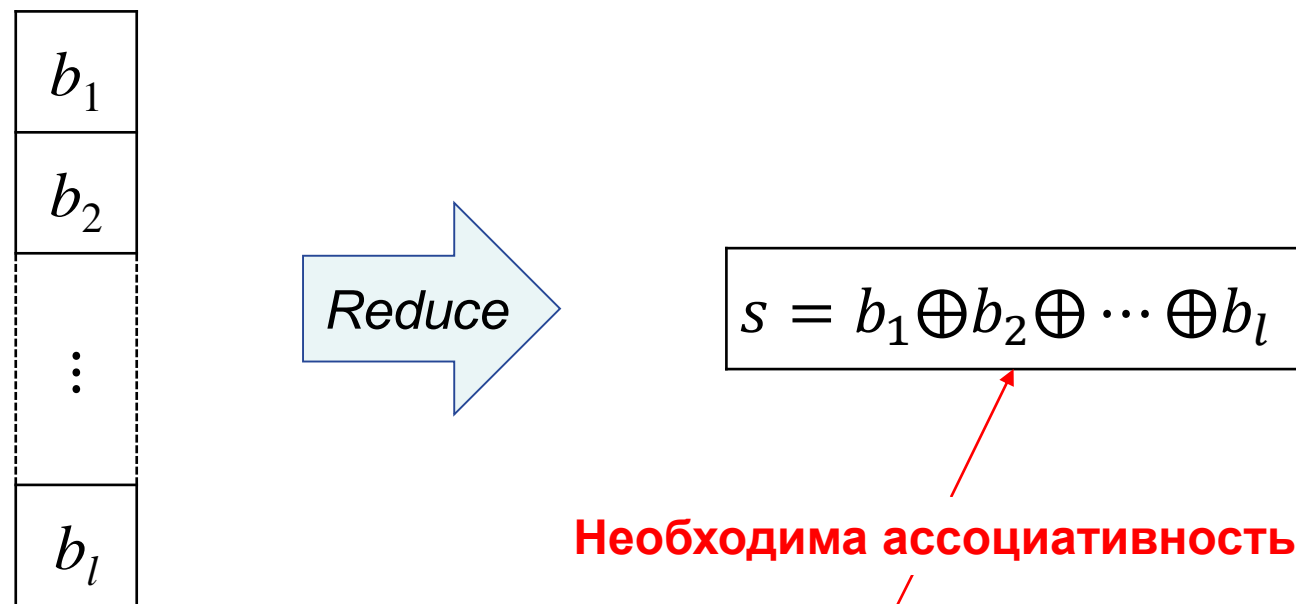
$$\text{Map}(F, [a_1, \dots, a_l]) = [F(a_1), \dots, F(a_l)]$$

# Функция высшего порядка

## *Reduce*

---

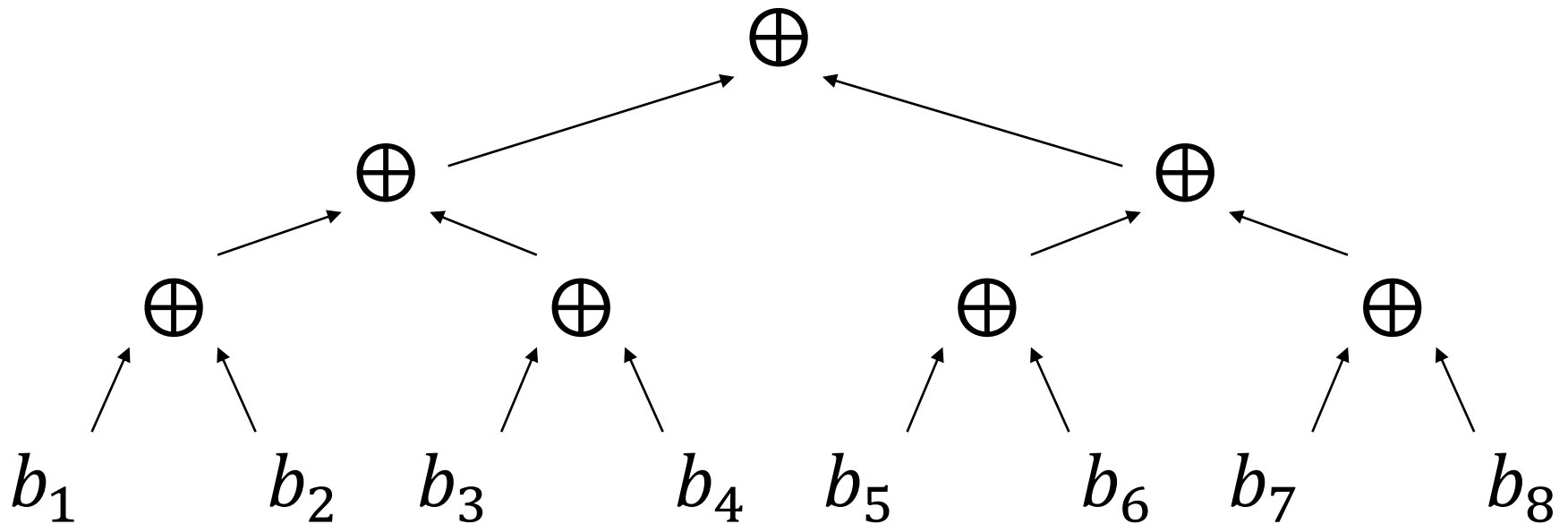
### Иерархический параллелизм



$$\text{Reduce}(\oplus, [b_1, \dots, b_l]) = b_1 \oplus \dots \oplus b_l$$

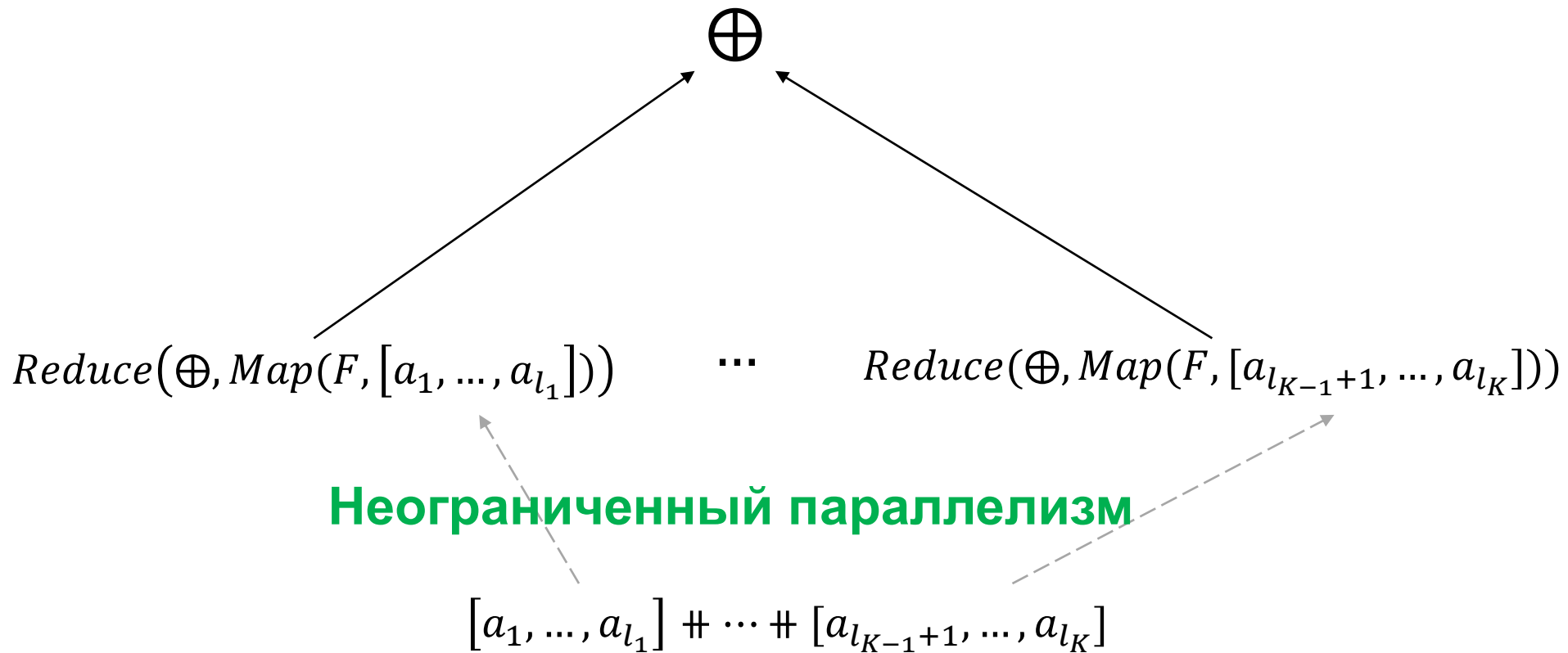
# Иерархический параллелизм

---



# Схема распараллеливания *Map & Reduce*

---



# Шаблон последовательного алгоритма в модели BSF

1.  $i := 0; \text{Input}(A, x_0)$
2.  $B := \text{Map}(F_{x_i}, A)$
3.  $s := \text{Reduce}(\oplus, B)$
4.  $x_{i+1} := \text{Compute}(x_i, s); i := i + 1$
5. **if**  $\text{StopCond}(x_{i-1}, x_i)$  **go to** 7
6. **go to** 2
7.  $\text{Output}(x_i); \text{stop}$

$i$	- номер итерации
$A \in [\mathcal{A}]$	- список исходных элементов данных
$x_0$	- начальное приближение
$F_x: \mathcal{A} \rightarrow \mathcal{B}$	- параметризованная функция
$B \in [\mathcal{B}]$	- список результирующих элементов
$\oplus$	- ассоциативная операция
$x_i$	- $i$ -тое приближение

# Map & Reduce – не прокрустово ложе!

- **Операция объединения списков  $\#$**

- **ассоциативность:**

$$([a_1, \dots, a_i] \# [b_1, \dots, b_j]) \# [c_1, \dots, c_l] = [a_1, \dots, a_i] \# ([b_1, \dots, b_j] \# [c_1, \dots, c_l])$$

- **Операция объединения мультимножеств  $\uplus$**

- **ассоциативность:**

$$(\{a_1, \dots, a_i\} \uplus \{b_1, \dots, b_j\}) \uplus \{c_1, \dots, c_l\} = \{a_1, \dots, a_i\} \uplus (\{b_1, \dots, b_j\} \uplus \{c_1, \dots, c_l\})$$

- **коммутативность:**

$$\{a_1, \dots, a_i\} \uplus \{b_1, \dots, b_j\} = \{b_1, \dots, b_j\} \uplus \{a_1, \dots, a_i\}$$

- **Операция объединения множеств  $\cup$**

- **ассоциативность:**

$$(\{a_1, \dots, a_i\} \cup \{b_1, \dots, b_j\}) \cup \{c_1, \dots, c_l\} = \{a_1, \dots, a_i\} \cup (\{b_1, \dots, b_j\} \cup \{c_1, \dots, c_l\})$$

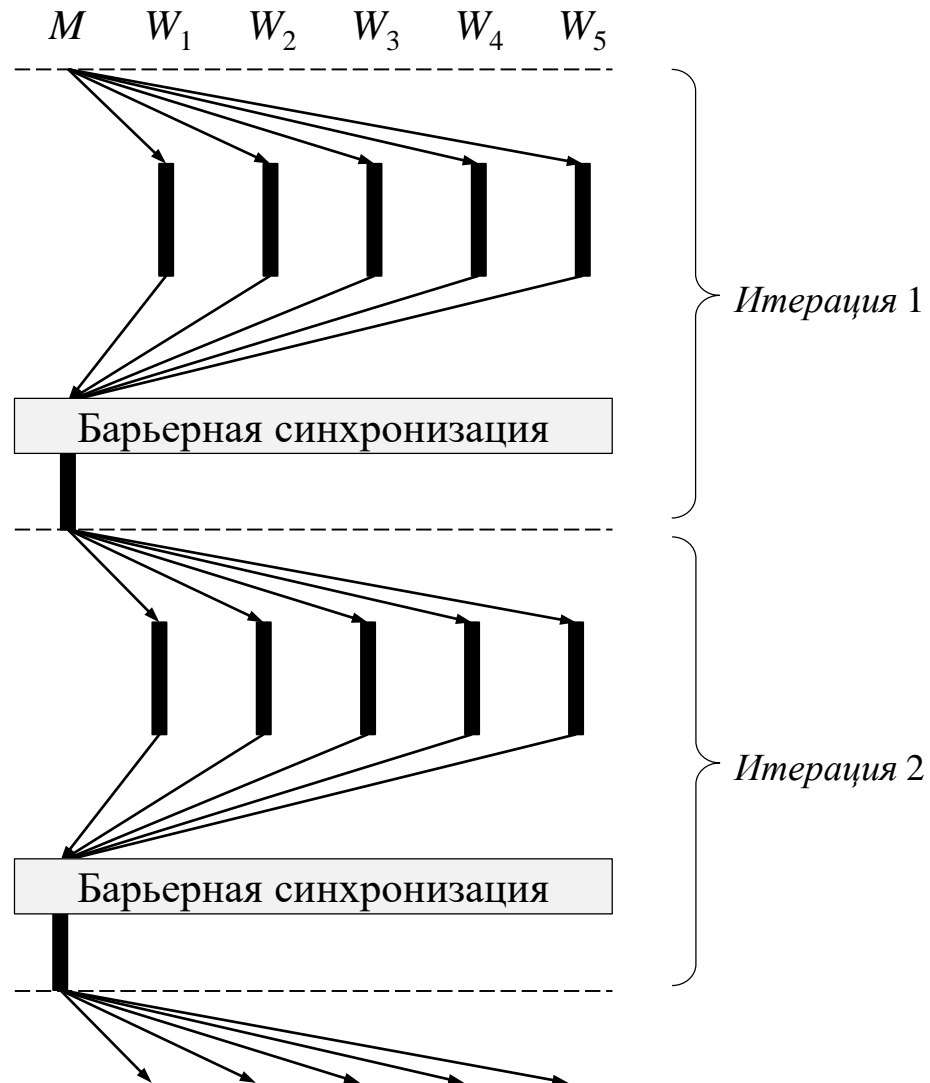
- **коммутативность:**

$$\{a_1, \dots, a_i\} \cup \{b_1, \dots, b_j\} = \{b_1, \dots, b_j\} \cup \{a_1, \dots, a_i\}$$

- **идемпотентность:**

$$\{a_1, \dots, a_i\} \cup \{b_1, \dots, b_j\} \cup \{b_1, \dots, b_j\} = \{a_1, \dots, a_i\} \cup \{b_1, \dots, b_j\}$$

# Синхронизация параллельной обработки и в модели *BSF*



# Шаблон параллельного алгоритма в модели BSF

Шаг	Мастер	Рабочий j (j=1,...,K)
1.	$i := 0; \text{Input}(x_0)$	$\text{Input}(A^{[j]})$
2.	$\text{SendToWorkers}(x_i)$	$\text{RecvFromMaster}(x_i)$ <hr style="border-top: 1px dashed black;"/> $B^{[j]} := \text{Map}(F_{x_i}, A^{[j]})$
3.	$\text{RecvFromWorkers}([s^{(1)}, \dots, s^{(K)}])$ <hr style="border-top: 1px dashed black;"/> $s := \text{Reduce}(\oplus, [s^{(1)}, \dots, s^{(K)}])$	$s^{(j)} := \text{Reduce}(\oplus, B^{[j]})$ <hr style="border-top: 1px dashed black;"/> $\text{SendToMaster}(s^{(j)})$
4.	$x_{i+1} := \text{Compute}(x_i, s); i := i + 1$	
5.	<b>if</b> $\text{StopCond}(x_{i-1}, x_i)$ <b>go to 7</b>	
6.	<b>go to 2</b>	<b>go to 2</b>
7.	$\text{Output}(x_i); \text{stop}$	

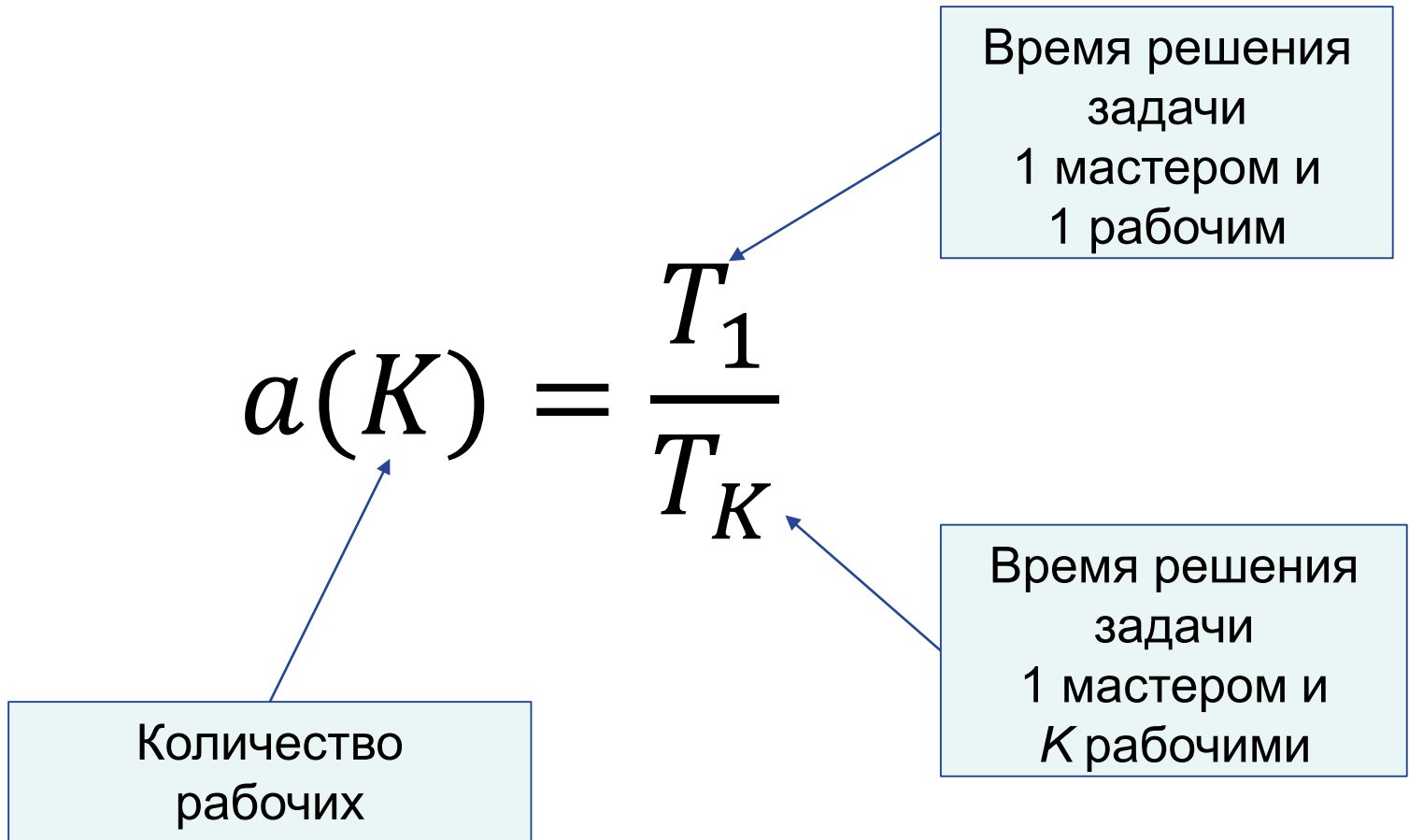


# Стоимостная метрика модели BSF

---

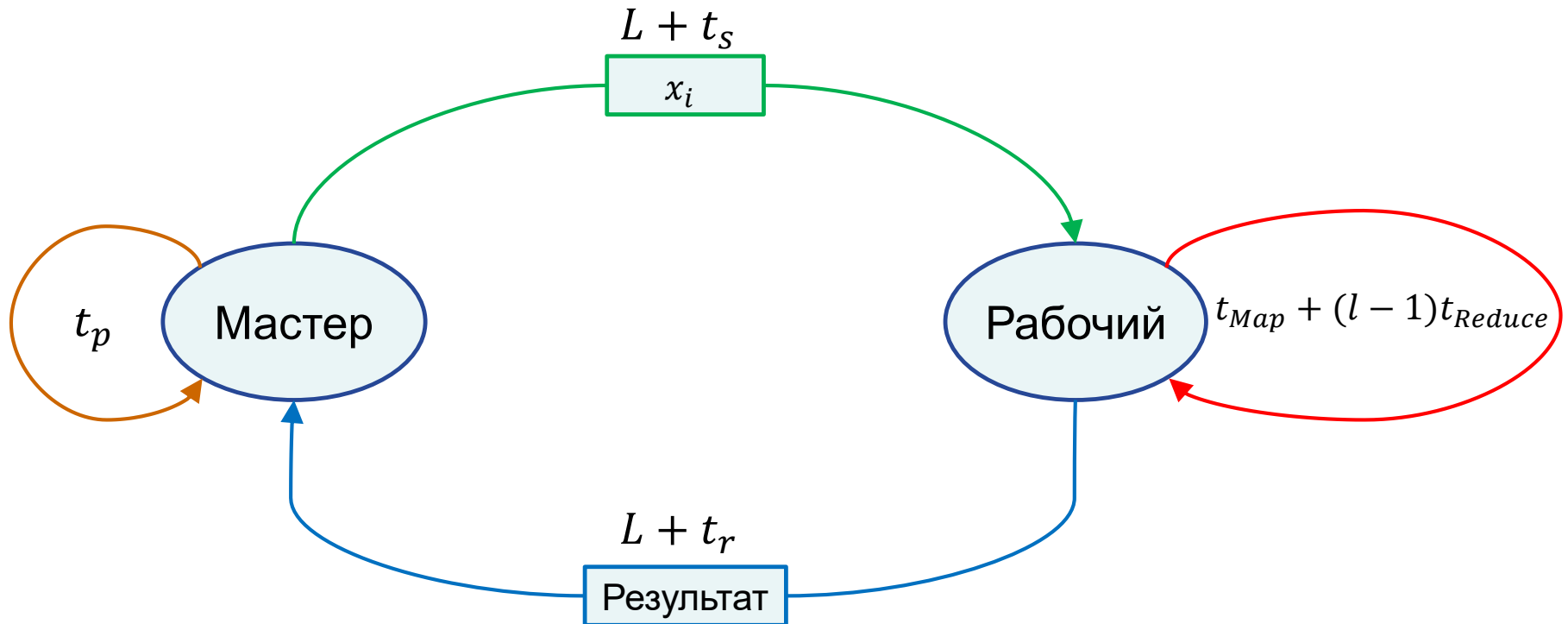
- $K$  – количество рабочих
- $l$  – длина обрабатываемого списка
- $L$  – латентность (время посылки сообщения длиной в 1 байт)
- $t_s$  – время передачи сообщения от мастера рабочему (без учета латентности)
- $t_r$  – время передачи сообщения от рабочего мастеру (без учета латентности)
- $t_{Map}$  – время выполнения функции Map для всего списка исходных данных
- $t_{Reduce}$  – время выполнения операции  $\oplus$
- $t_p$  – время на обработку результатов итерации

# Ускорение в модели BSF



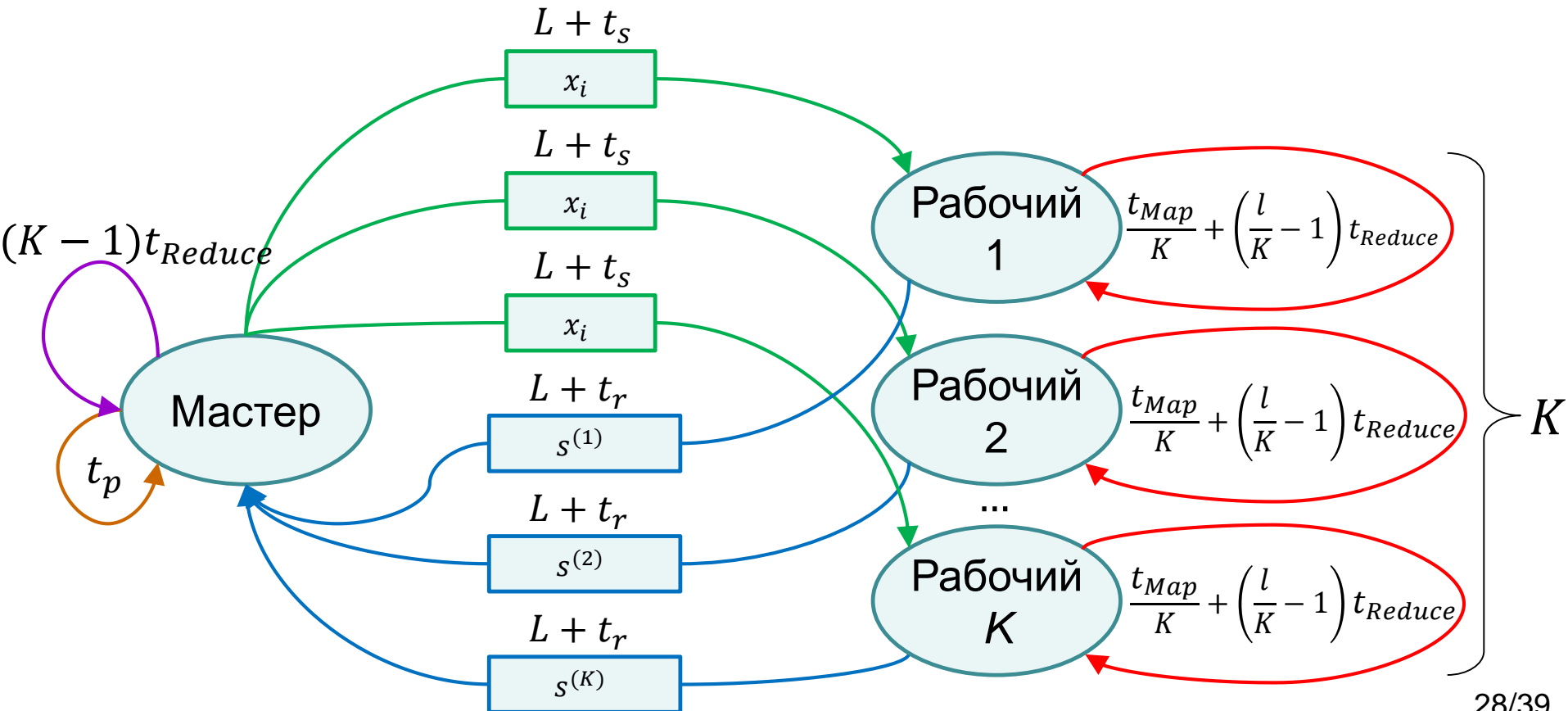
# Оценка времени $T_1$

$$T_1 = L + t_s + t_{Map} + (l - 1)t_{Reduce} + L + t_r + t_p$$



# Оценка времени $T_K$

$$T_K = K(L + t_s) + \frac{t_{Map}}{K} + \left(\frac{l}{K} - 1\right) t_{Reduce} + K(L + t_r) + (K - 1)t_{Reduce} + t_p$$

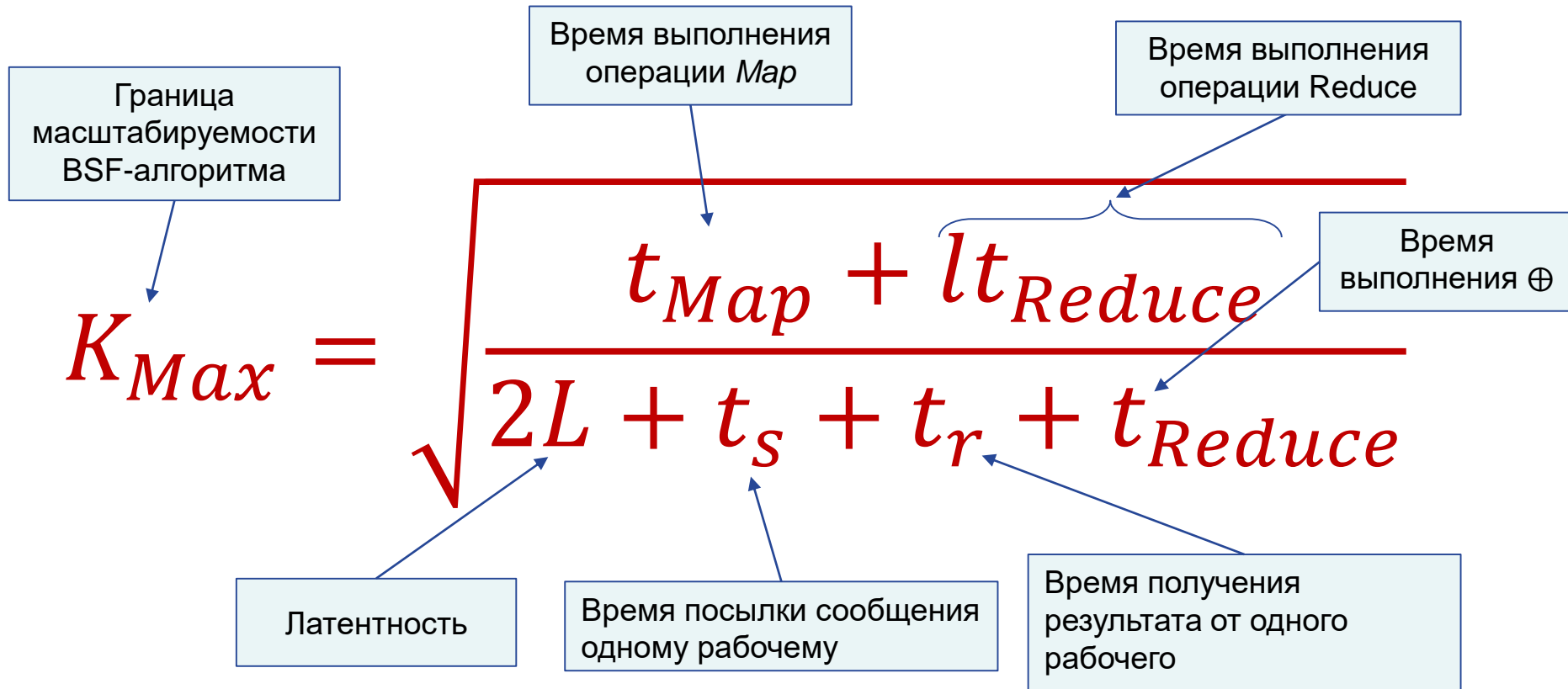


# Ускорение для модели BSF

---

$$a_{BSF}(K) = \frac{2L + t_s + t_r + t_p + t_{Map} + lt_{Reduce}}{K(2L + t_s + t_r + t_{Reduce}) + \frac{(t_{Map} + lt_{Reduce})}{K} - t_{Reduce} + t_p}$$

# Теорема о границе масштабируемости BSF-алгоритма



# Алгоритм Якобі для приближенного решения СЛАУ

---

$$Ax = b$$

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \quad x = (x_1, \dots, x_n) \quad b = (b_1, \dots, b_n)$$

$$C = \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{pmatrix} \quad c_{ij} = \begin{cases} -\frac{a_{ij}}{a_{ii}}, \forall j \neq i \\ 0, \forall j = i \end{cases}$$

$$d = (d_1, \dots, d_n) \quad d_i = b_i / a_{ii}$$

$$x^{(k+1)} = Cx^{(k)} + d$$

# Функция для *Map*

---

$$F_x(j) = (c_{1j}x_j, \dots, c_{nj}x_j) = x_j \begin{pmatrix} c_{1j} \\ \vdots \\ c_{nj} \end{pmatrix}^T$$

$j$  – номер столбца матрицы  $C$



# Алгоритм Якобі над списками

---

1.  $Input(C, d); k := 0; x^{(0)} := d$
2.  $[g^1, \dots, g^n] := Map(F_{x^{(k)}}, [1, \dots, n])$
3.  $g := Reduce(+, [g^1, \dots, g^n])$
4.  $x^{(k+1)} := g + d; k := k + 1$
5. **if**  $\|x^{(k)} - x^{(k-1)}\|^2 < \varepsilon$  **go to 7**
6. **go to 2**
7.  $Output(x^{(k)});$  **stop**

# BSF-оценки для алгоритма Jacobi

- $\tau_{op}$  - время выполнения одной операции с плавающей точкой
- $\tau_{tr}$  - время пересылки вещественного числа (без учета латентности)
- $n$  - количество уравнений
- $L$  - латентность

Ускорение для модели BSF

$$a_{BSF}(K) = \frac{2L + t_s + t_r + t_p + t_{Map} + lt_{Reduce}}{K(2L + t_s + t_r + t_{Reduce}) + \frac{(t_{Map} + lt_{Reduce})}{K} - t_{Reduce} + t_p}$$

28/39

$$\Rightarrow a_{Jacobi}(K) = \frac{2(L + \tau_{tr}n) + \tau_{op}n(3n - K + 5)}{K^2 (2(L + \tau_{op}n) + 3\tau_{op}n(n + K))}$$

Теорема о границе масштабируемости BSF-алгоритма

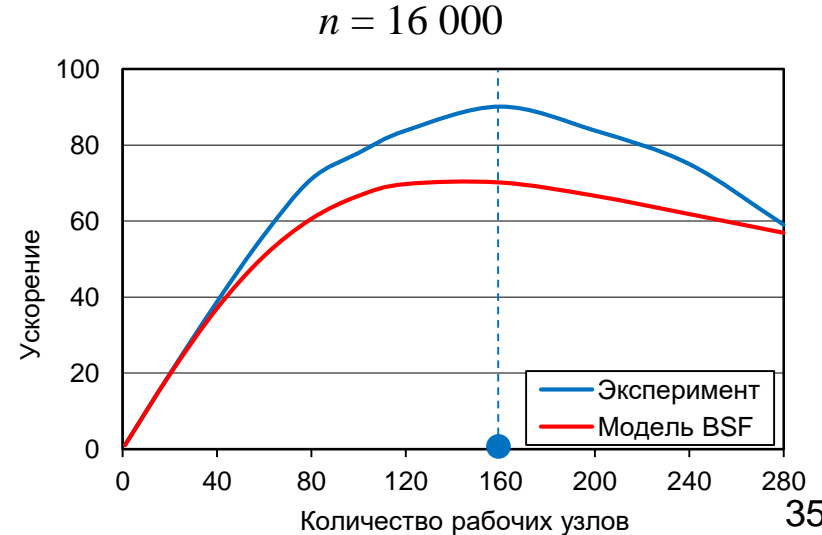
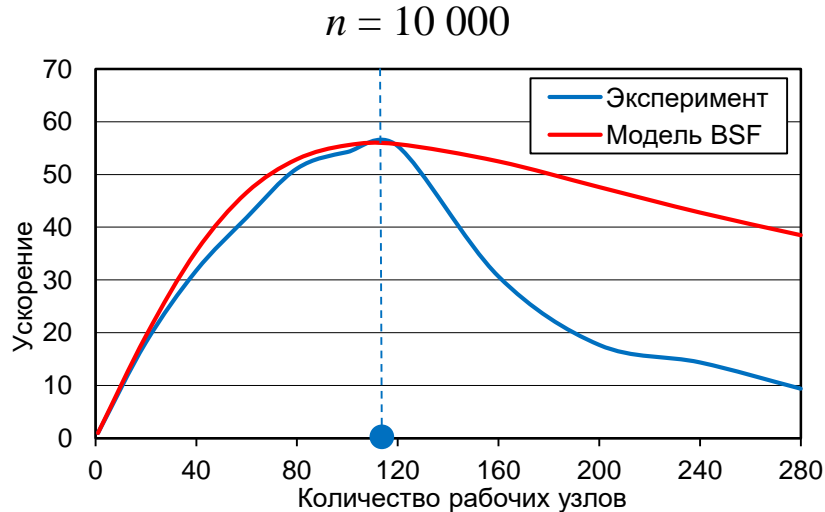
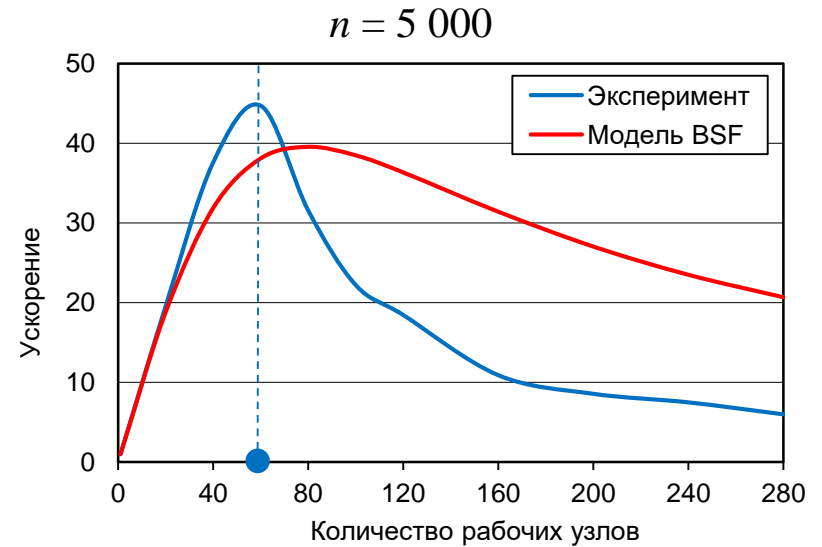
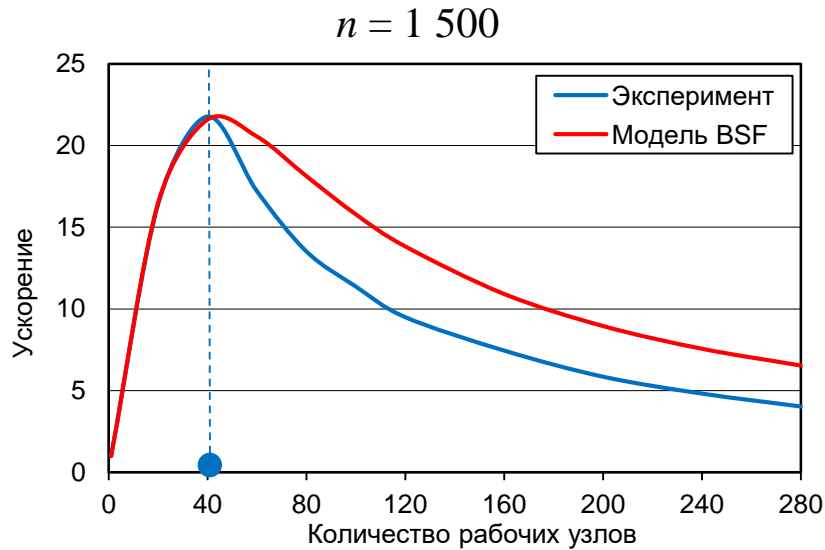
$$K_{Max} = \sqrt{\frac{t_{Map} + lt_{Reduce}}{2L + t_s + t_r + t_{Reduce}}}$$

30/39

$$\Rightarrow K_{Jacobi}^{Max} = O(\sqrt{n})$$

# Ускорение алгоритма Якоби-MR:

## теория и практика



# Другие верификации модели BSF

---

- *Sokolinskaya I., Sokolinsky L.B.* Scalability Evaluation of NSLP Algorithm for Solving Non-Stationary Linear Programming Problems on Cluster Computing Systems // Supercomputing. RuSCDays 2017. Communications in Computer and Information Science. 2017. Vol. 793. P. 40-53. DOI: 10.1007/978-3-319-71255-0\_4
- *Sokolinskaya I.M., Sokolinsky L.B.* Scalability Evaluation of Cimmino Algorithm for Solving Linear Inequality Systems on Multiprocessors with Distributed Memory // Supercomputing Frontiers and Innovations. 2018. Vol. 5, No. 2. P. 11-22. DOI: 10.14529/jsfi180202
- Ежова Н.А., Соколинский Л.Б. Верификация модели параллельных вычислений BSF-MR на примере гравитационной задачи // Параллельные вычислительные технологии (ПаВТ'2019). Короткие статьи и описания плакатов XIII Международной научной конференции. Челябинск: Издательский центр ЮУрГУ, 2019. С. 239-250. URL: <http://omega.sp.susu.ru/pavt2019/short/007.pdf>

# Параллельный BSF-каркас

---

- C++
- MPI + OpenMP
- Инкрементная компиляция
- Исходные коды  
<https://github.com/leonid-sokolinsky/BSF-skeleton>

# BSF-Studio

## (в разработке)

---

- Облачный конструктор для быстрой разработки BSF-программ на языке C++ с использованием библиотек MPI и OpenMP
- Базируется на BSF-каркасе



Разработчик:  
Надежда Ежова  
(аспирант)

# Спасибо за внимание!

---

Вопросы?