Научный семинар по информационным технологиям
(Челябинск, 9 октября 2018 г.)

Расширение модели параллельных вычислений BSF

Надежда Александровна Ежова

научный руководитель:
д.ф.-м.н., профессор
Л.Б. Соколинский

Южно-Уральский государственный университет
(национальный исследовательский университет)

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 17-07-00352а, Правительства РФ в соответствии с Постановлением №211 от 16.03.2013 г. (соглашение № 02.A03.21.0011) и Министерства образования и науки РФ (государственное задание 2.7905.2017/8.9).

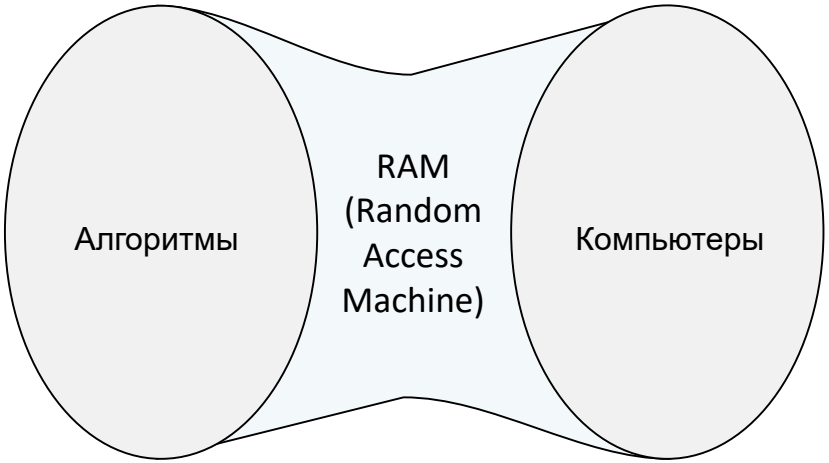
Bridging model (соединяющая модель)

- Bridging model – упрощенное, абстрактное описание компьютера
- Используется для разработки эффективных алгоритмов и программ
- Позволяет оценить реальную вычислительную сложность алгоритма и прогнозируемое быстродействие программы на различных вычислительных системах

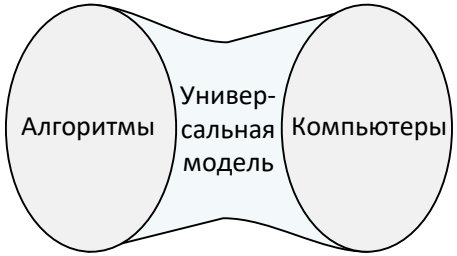
Valiant L.G. A bridging model for parallel computation // Commun. ACM. 1990. Vol. 33, № 8. P. 103–111.

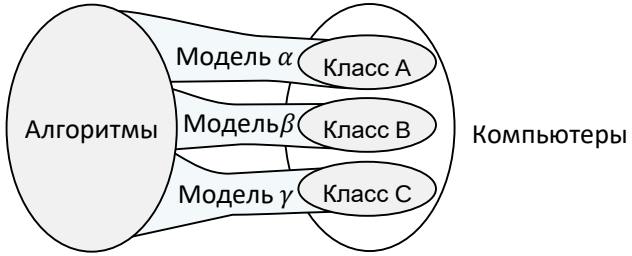
Модели, соединяющие алгоритмы и компьютеры

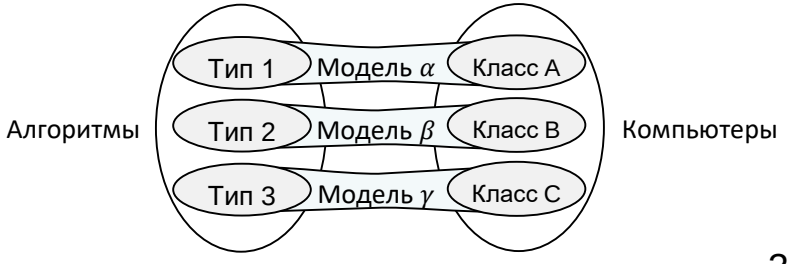
Последовательные алгоритмы и компьютеры



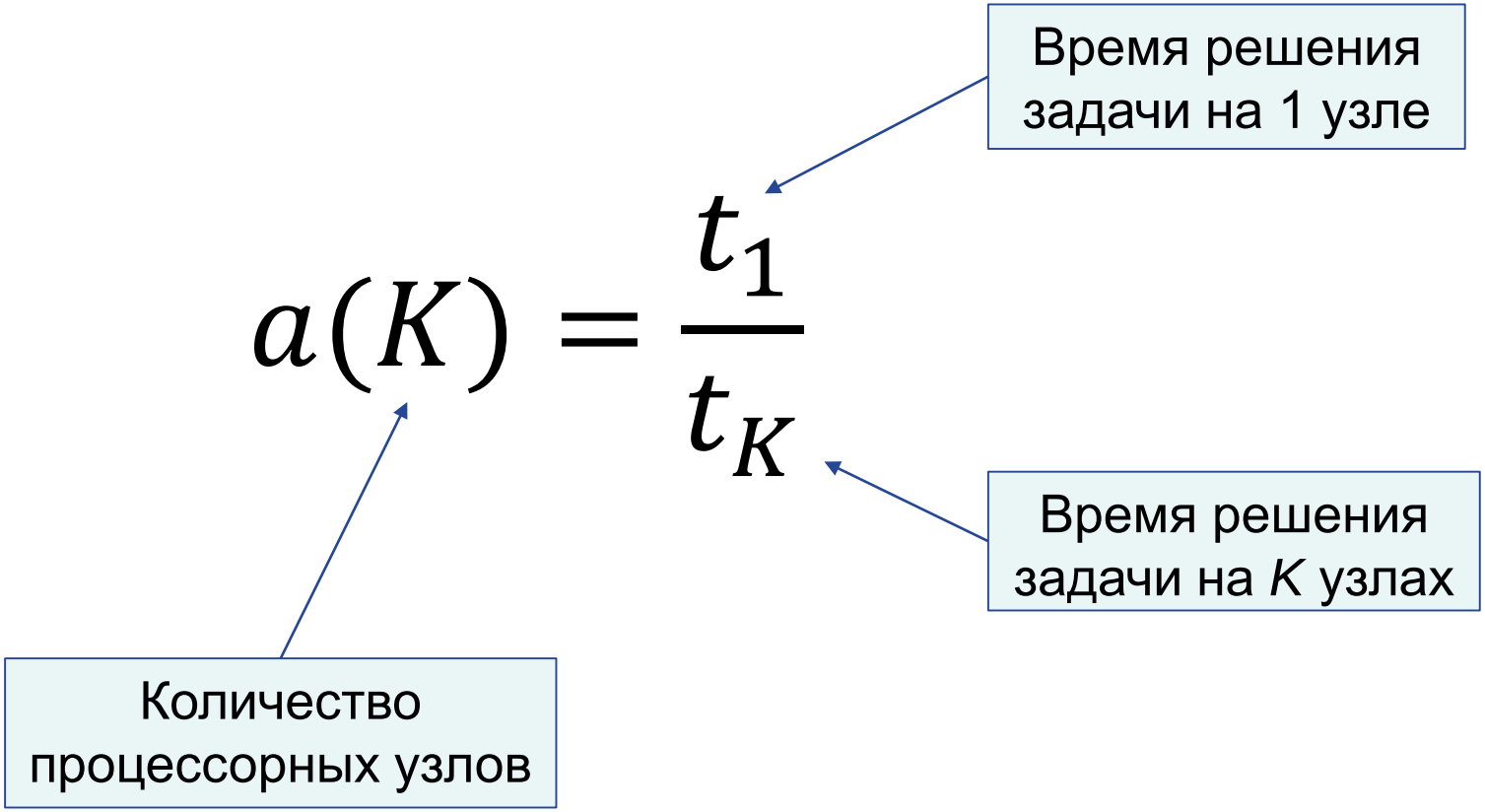
Параллельные алгоритмы и компьютеры



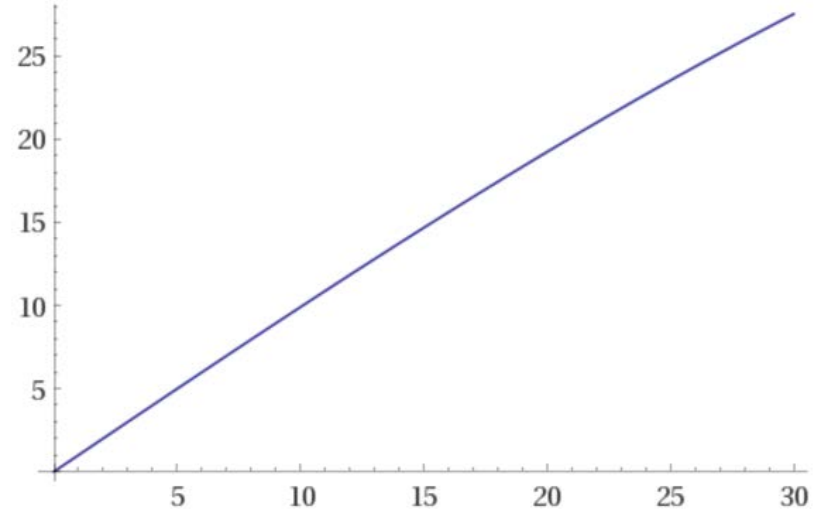


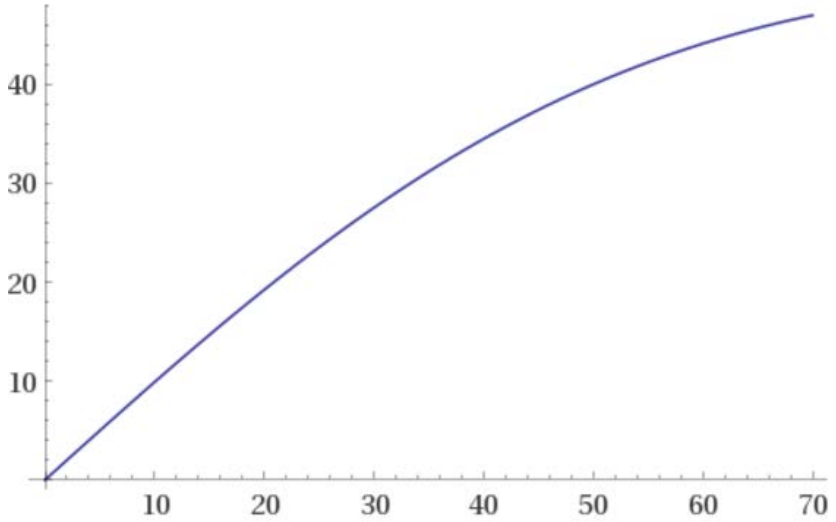


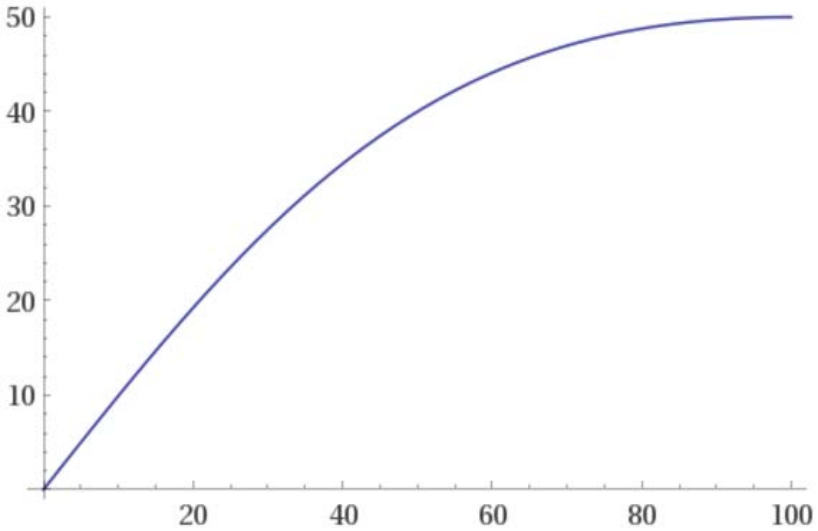
Ускорение – главная характеристика масштабируемого алгоритма

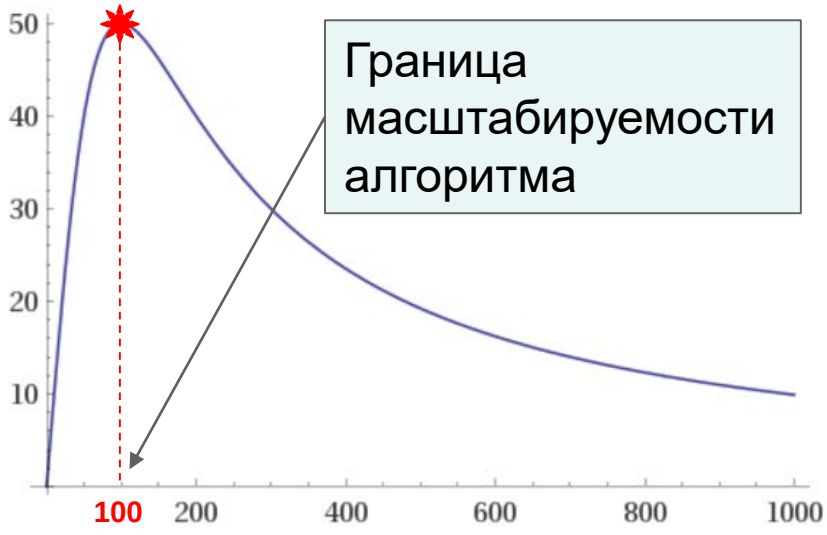


Кривые ускорения реальной задачи на кластерной вычислительной системе (ускорение / количество процессорных узлов)





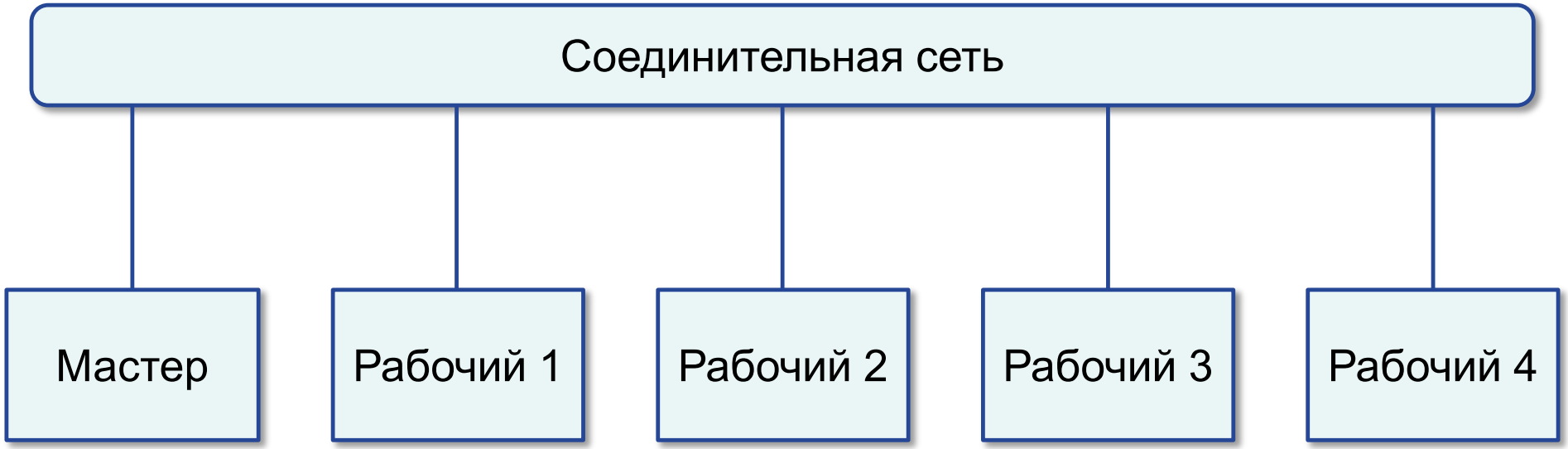




Модель параллельных вычислений BSF (Bulk-Synchronous Farm)

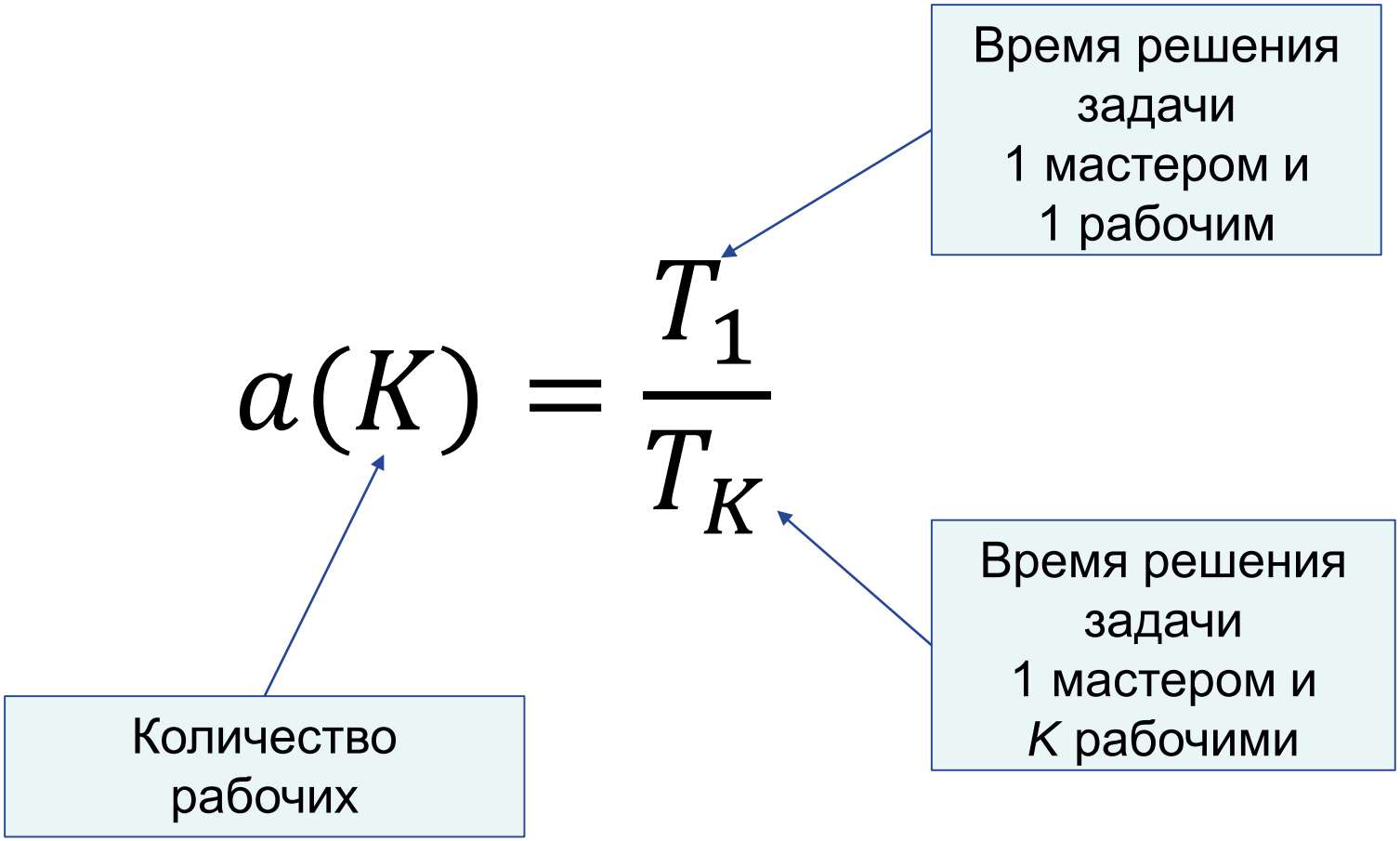
- *Модель BSF* – фреймворк (система правил и ограничений) для описания и анализа параллельных алгоритмов и программ
- Область применения:
  - Многопроцессорные системы с распределенной памятью
  - Параллельные итерационные алгоритмы с высокой вычислительной сложностью
- Позволяет предсказать:
  - границу масштабируемости параллельного алгоритма
  - ускорение параллельного алгоритма

BSF-компьютер



Процессорные узлы

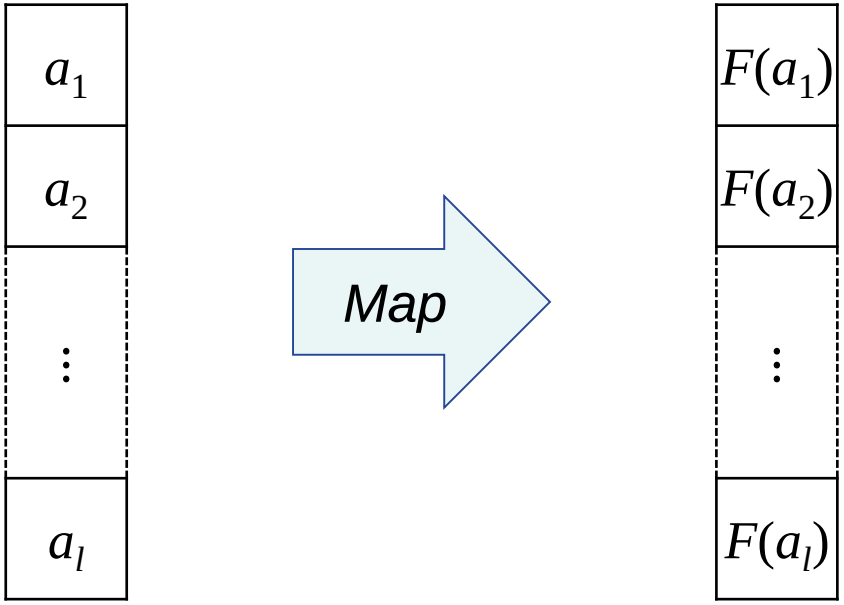
Ускорение в модели BSF



Представление алгоритма в виде операций над списками

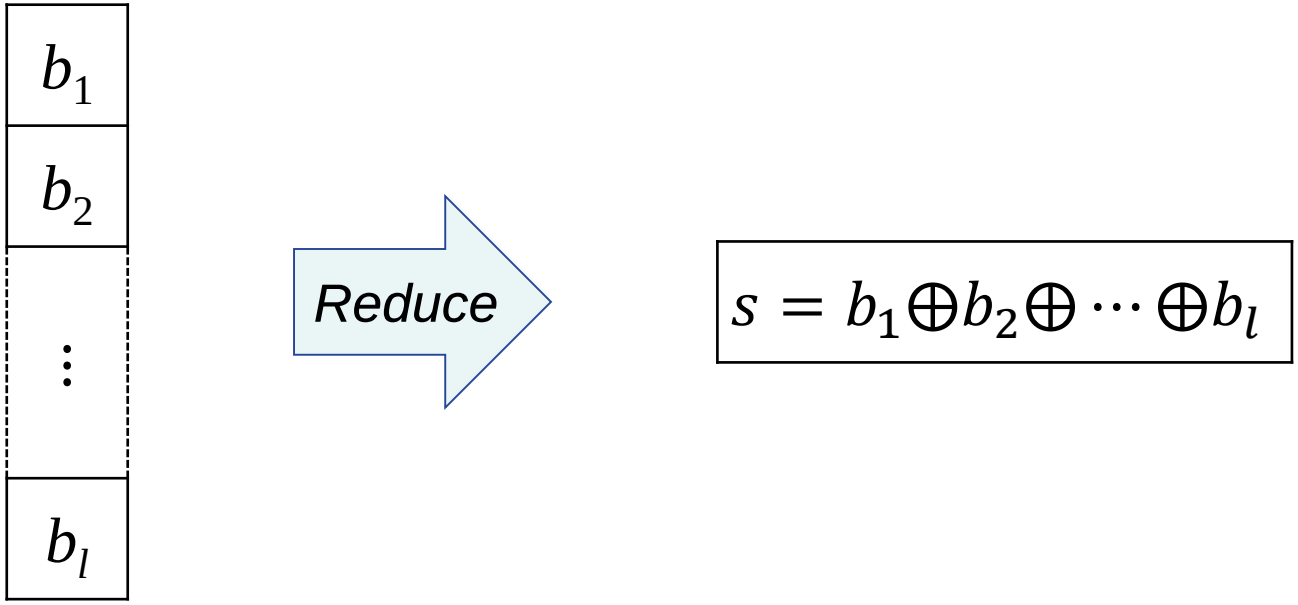
- Функции высшего порядка:
  - Map
  - Reduce

Функция высшего порядка *Map*



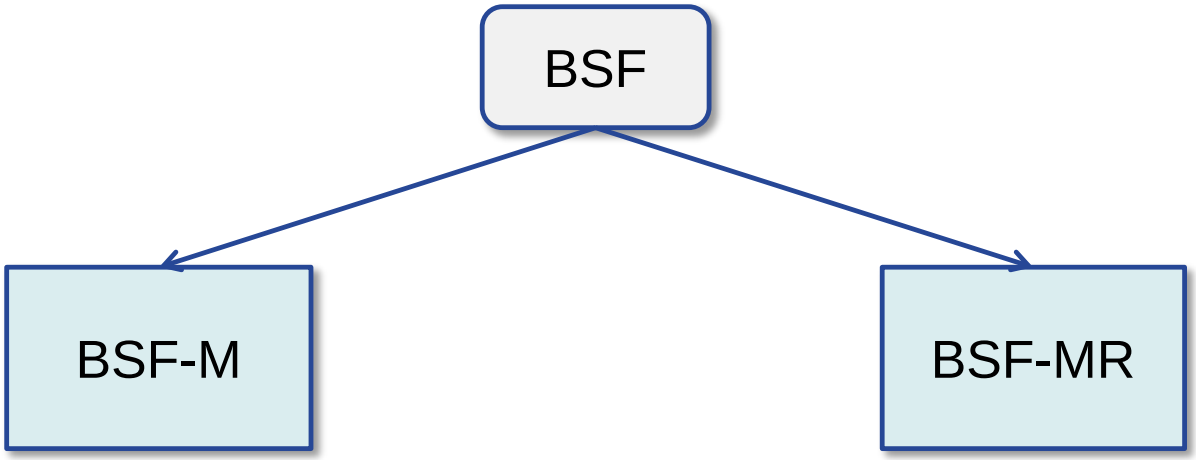
$$Map(F, [a_1, \ldots, a_l]) = [F(a_1), \ldots, F(a_l)]$$

Функция высшего порядка *Reduce*



$$Reduce(\oplus, [b_1, \ldots, b_l]) = b_1 \oplus \cdots \oplus b_l$$

Расширения модели BSF



Расширение BSF-M

Используется функция высшего порядка *Map*

Шаблон итерационного алгоритма в BSF-M

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

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

Шаблон параллельного алгоритма в BSF-M

Шаг	Мастер	Рабочий j (j=1,…,K)
1.	$i := 0; Input(A, x_0)$	$Input(A)$
2.	$SendToWorkers(x_i)$	$RecvFromMaster(x_i)$
		$B^{[j]} := Map(F_{x_i}, A^{[j]})$
3.	$RecvFromWorkers(B)$	$SendToMaster(B^{[j]})$
	$x_{i+1} := Comput(x_i, B)$	
4.	**if** $StopCond(x_{i+1}, x_i)$ **go to** 7	
5.	$i := i + 1$	
6.	**go to** 2	**go to** 2
7.	$Output(x_{i+1})$; **stop**	

Параметры BSF-M

K – количество рабочих

l – длина списка ($l = mK, m \in \mathbb{N}$)

t_s – время, затрачиваемое мастером на передачу сообщения одному рабочему (без учета латентности)

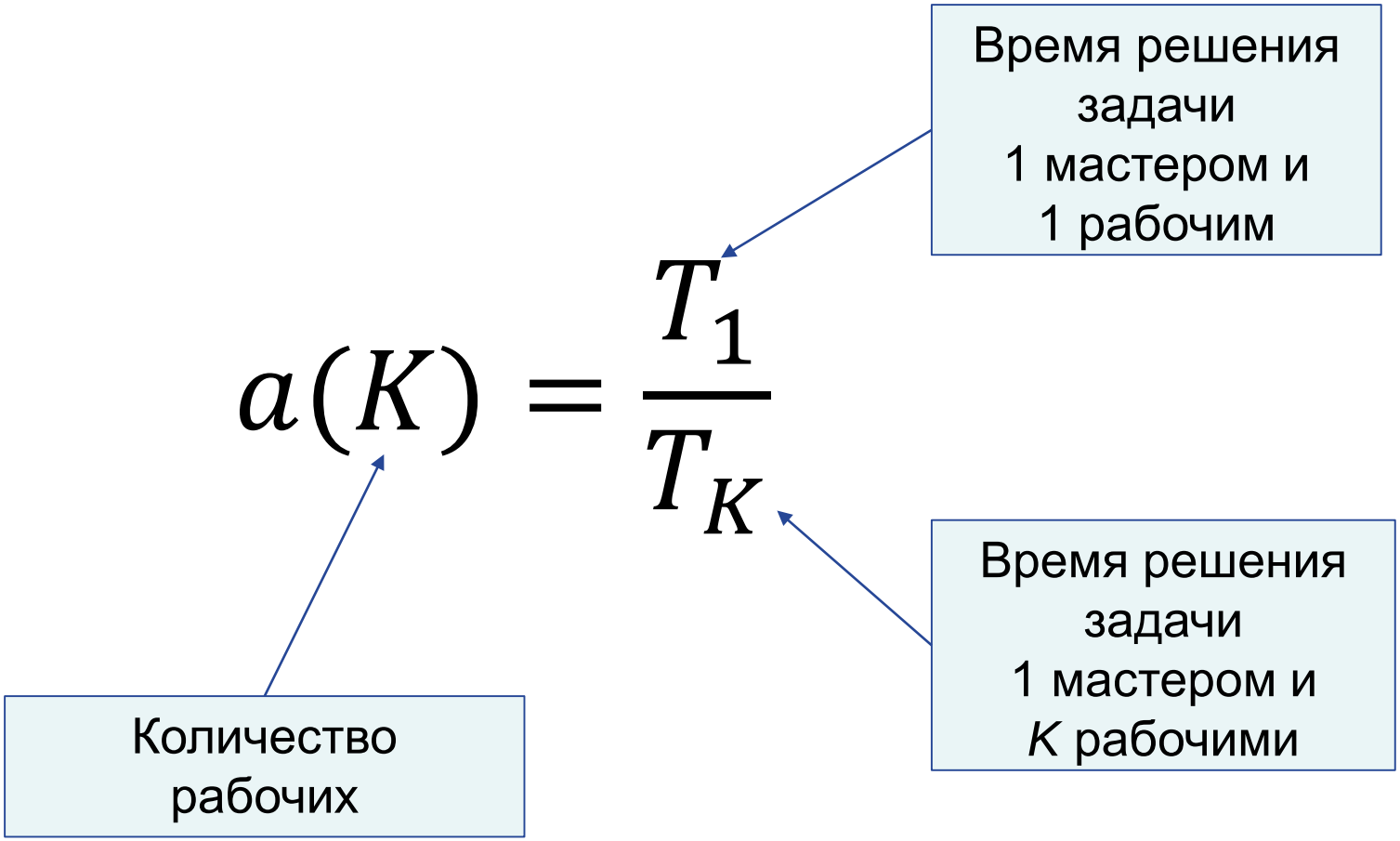
L – латентность (время посылки сообщения длиной в 1 байт)

t_w – время выполнения задания бригадой из одного рабочего в рамках одной итерации

t_R – время, затрачиваемое мастером на получение результатов от всех рабочих (без учета латентности)

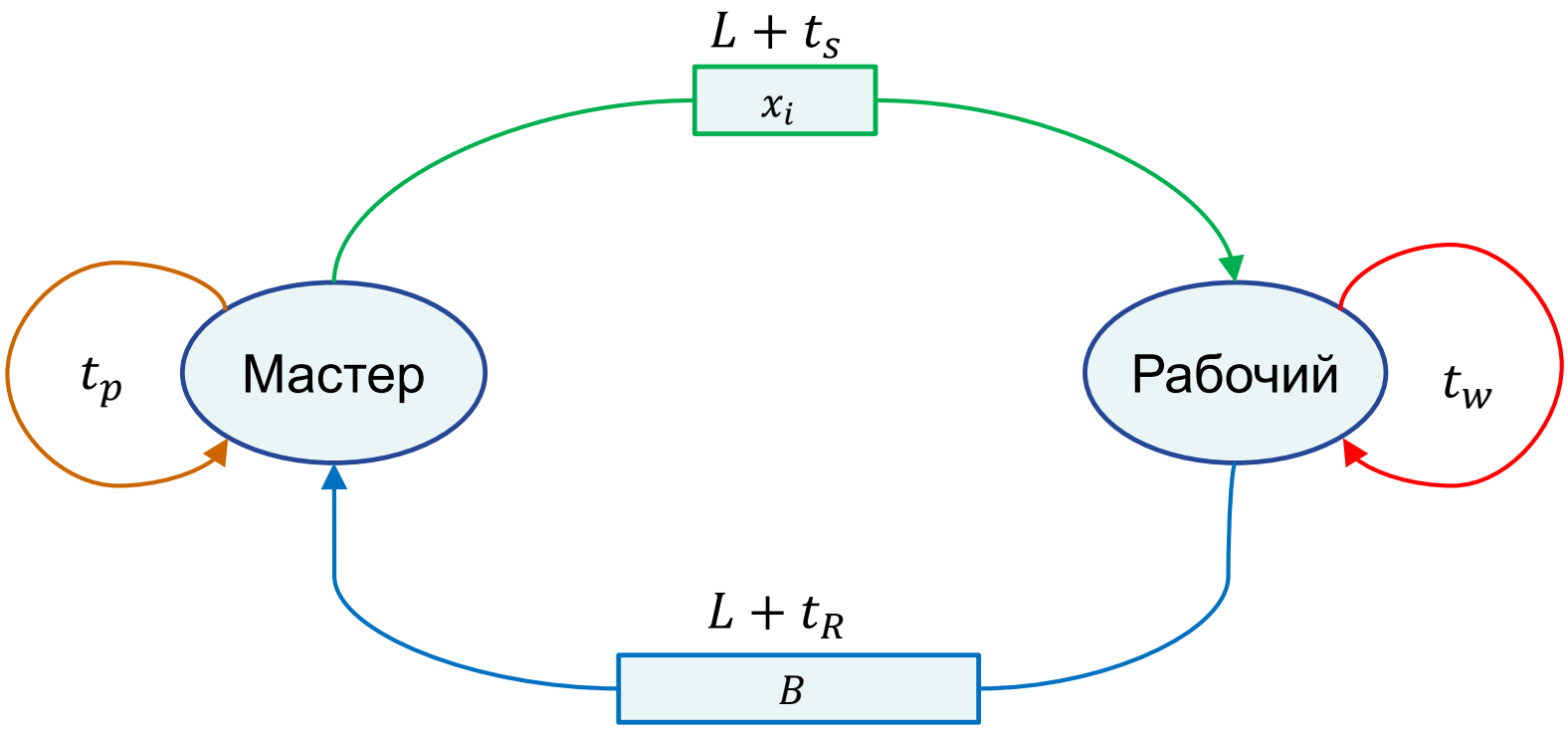
t_p – время, затрачиваемое мастером на обработку полученных результатов и проверку условия завершения

Ускорение в модели BSF-M



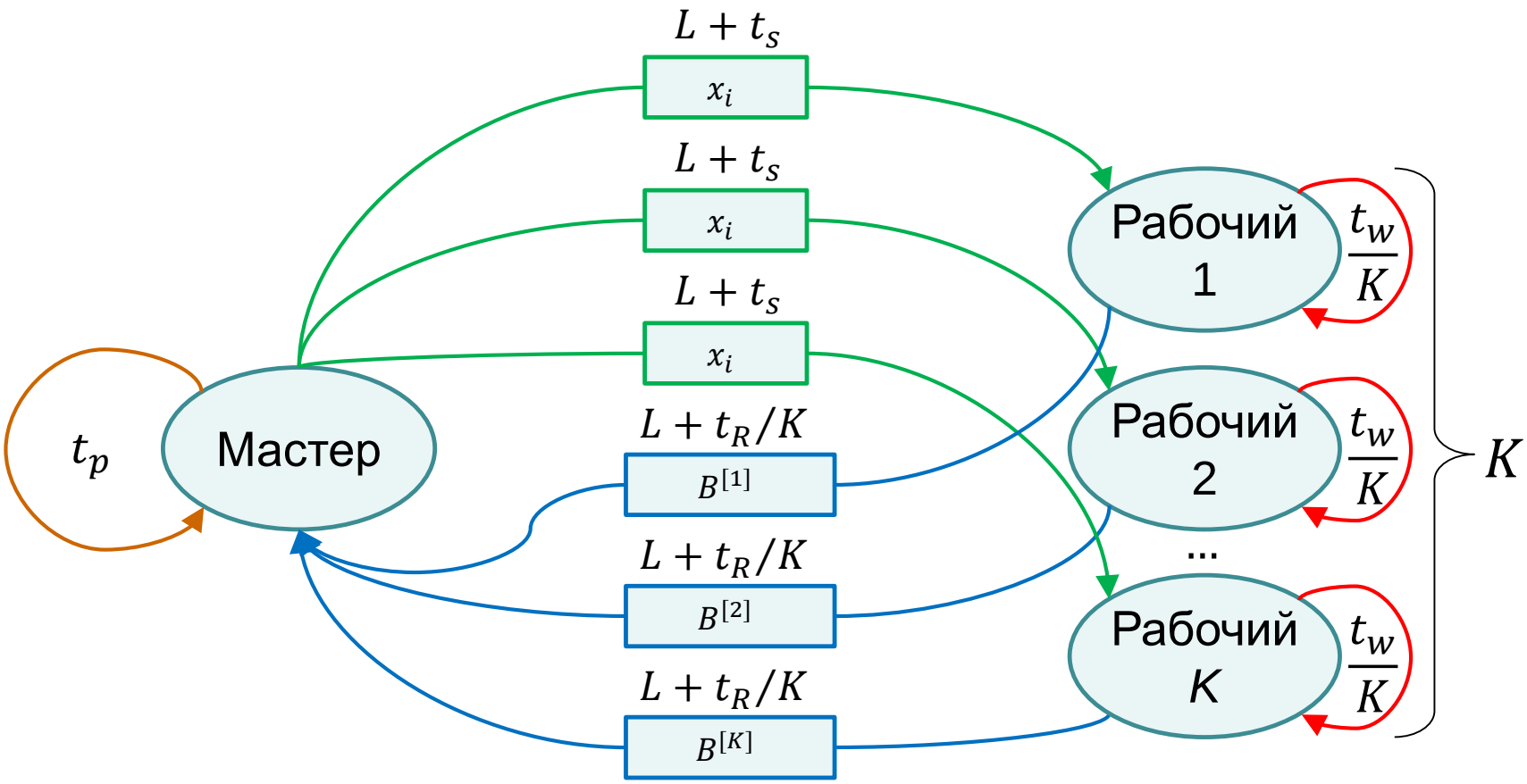
Время решения задачи системой из одного мастера и одного рабочего для BSF-M (сек.)

$$T_1 = L + t_s + t_w + L + t_R + t_p$$



Время решения задачи системой из одного мастера и K рабочих для BSF-M (сек.)

$$T_K = K(L + t_s) + t_w/K + K(L + t_R/K) + t_p$$

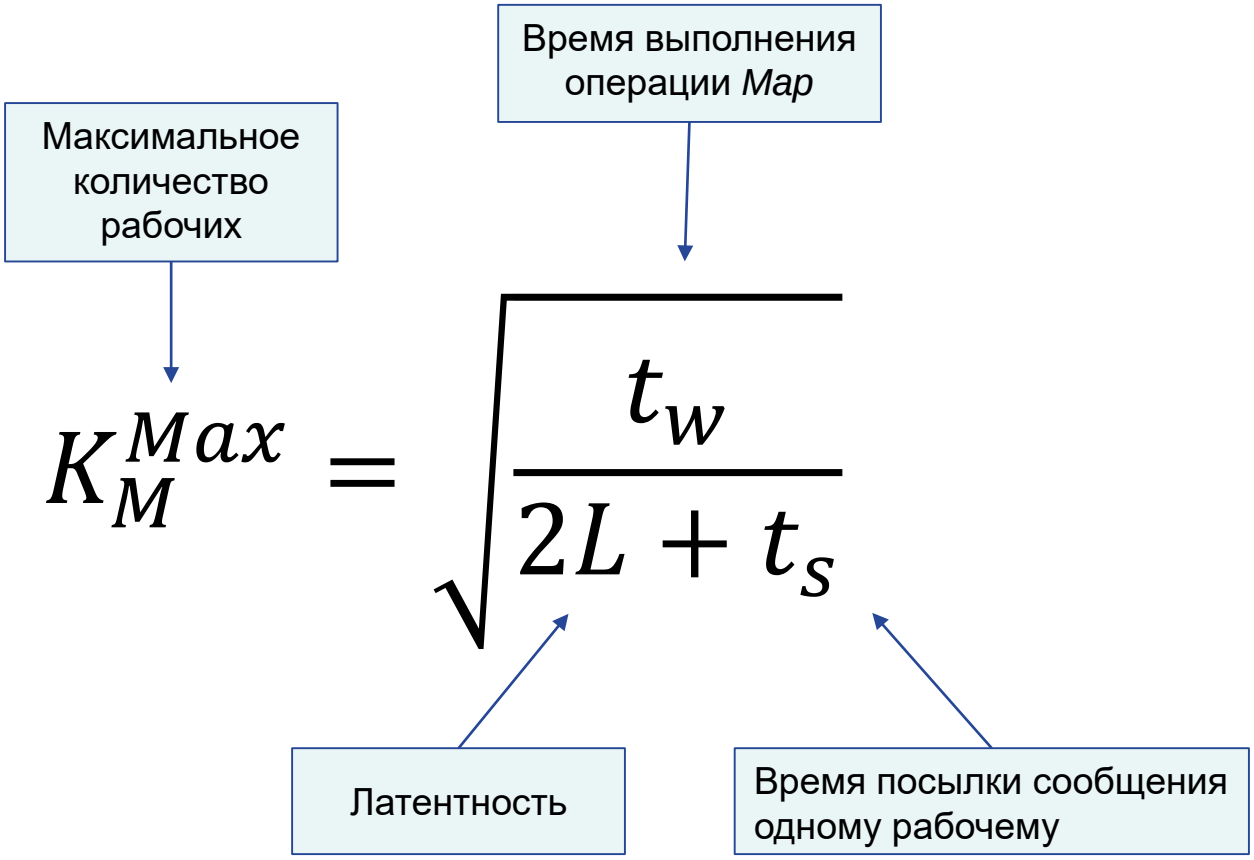


Ускорение для BSF-M

$$a(K) = \frac{T_1}{T_K}$$

$$a_{BSF-M}(K) = \frac{2L + t_s + t_R + t_p + t_w}{K^2(2L + t_s) + K\left(t_R + t_p\right) + t_w}$$

Граница масштабируемости для BSF-M



Расширение BSF-MR

Используются функции высшего порядка *Map* и *Reduce*

Шаблон итерационного алгоритма в BSF-MR

1. $i := 0; Input(A, x_0)$
2. $B := Map(F_{x_i}, A)$
3. $s := Reduce(\oplus, B)$
4. $x_{i+1} := Comput(x_i, s)$
5. **if** $StopCond(x_{i+1}, x_i)$ **go to** 8
6. $i := i + 1$
7. **go to** 2
8. $Output(x_{i+1})$; **stop**

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

Шаблон параллельного алгоритма в BSF-MR

Шаг	Мастер	Рабочий j (j=1,…,K)
1.	$i := 0; Input(A, x_0)$	$Input(A)$
2.	$SendToWorkers(x_i)$	$RecvFromMaster(x_i)$
		$B^{[j]} := Map\left(F_{x_i}, A^{[j]}\right)$
3.		$s^{(j)} := Reduce\left(\oplus, B^{[j]}\right)$
	$RecvFromWorkers([s^{(1)}, \dots, s^{(K)}])$	$SendToMaster(s^{(j)})$
	$s := Reduce\left(\oplus, [s^{(1)}, \dots, s^{(K)}]\right)$	
4.	$x_{i+1} := Comput(x_i, s)$	
5.	**if** $StopCond(x_{i+1}, x_i)$ **go to** 8	
6.	$i := i + 1$	
7.	**go to** 2	**go to** 2
8.	$Output(x_{i+1})$; **stop**	

Параметры BSF-MR

K – количество рабочих

l – длина списка ($l = mK, m \in \mathbb{N}$)

t_s – время, затрачиваемое мастером на передачу сообщения одному рабочему (без учета латентности)

L – латентность (время посылки сообщения длиной в 1 байт)

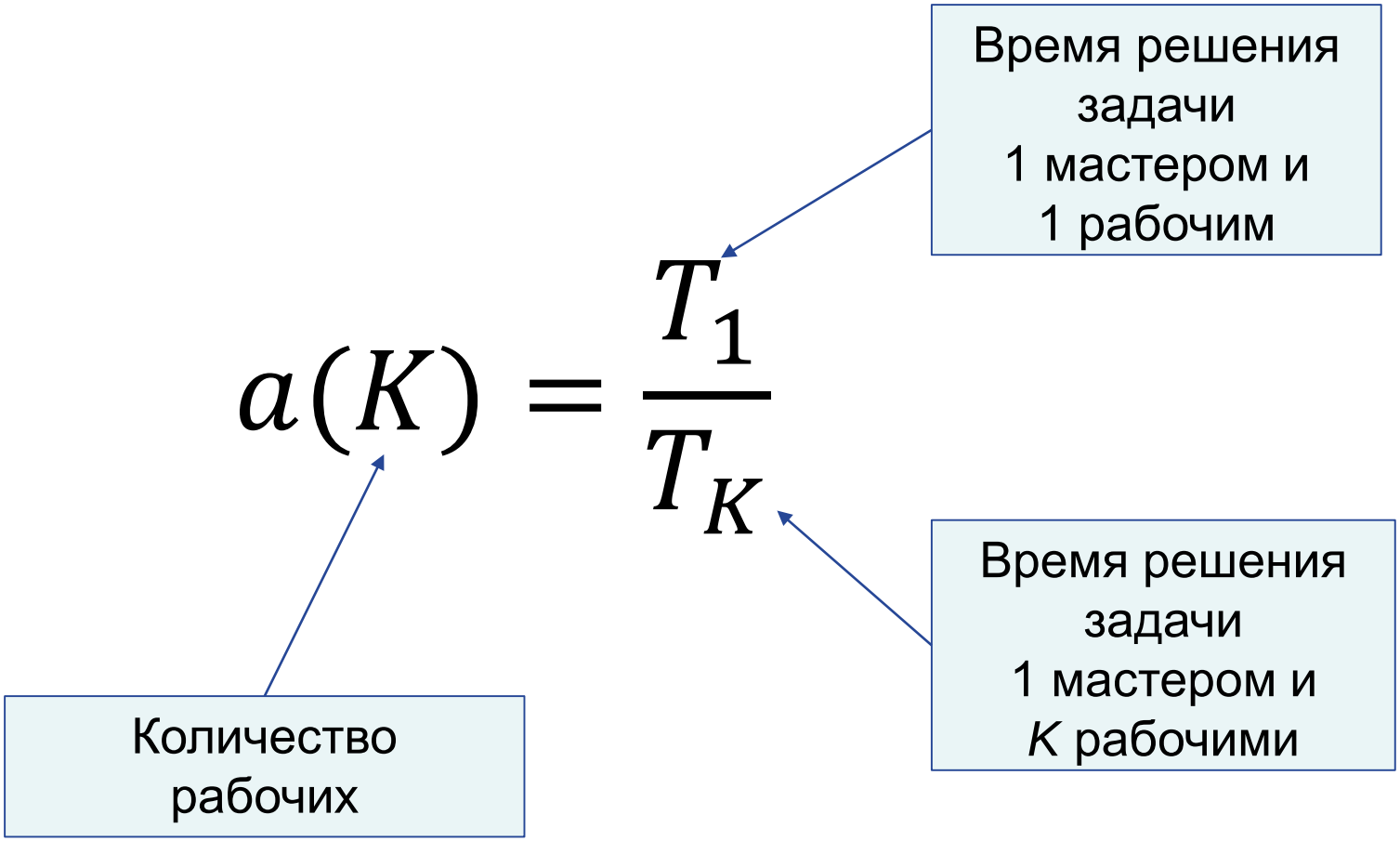
t_w – время выполнения задания бригадой из одного рабочего в рамках одной итерации

t_r – время, затрачиваемое мастером на получение сообщения от одного рабочего (без учета латентности)

t_a – время, необходимое для выполнения одной операции $\oplus$

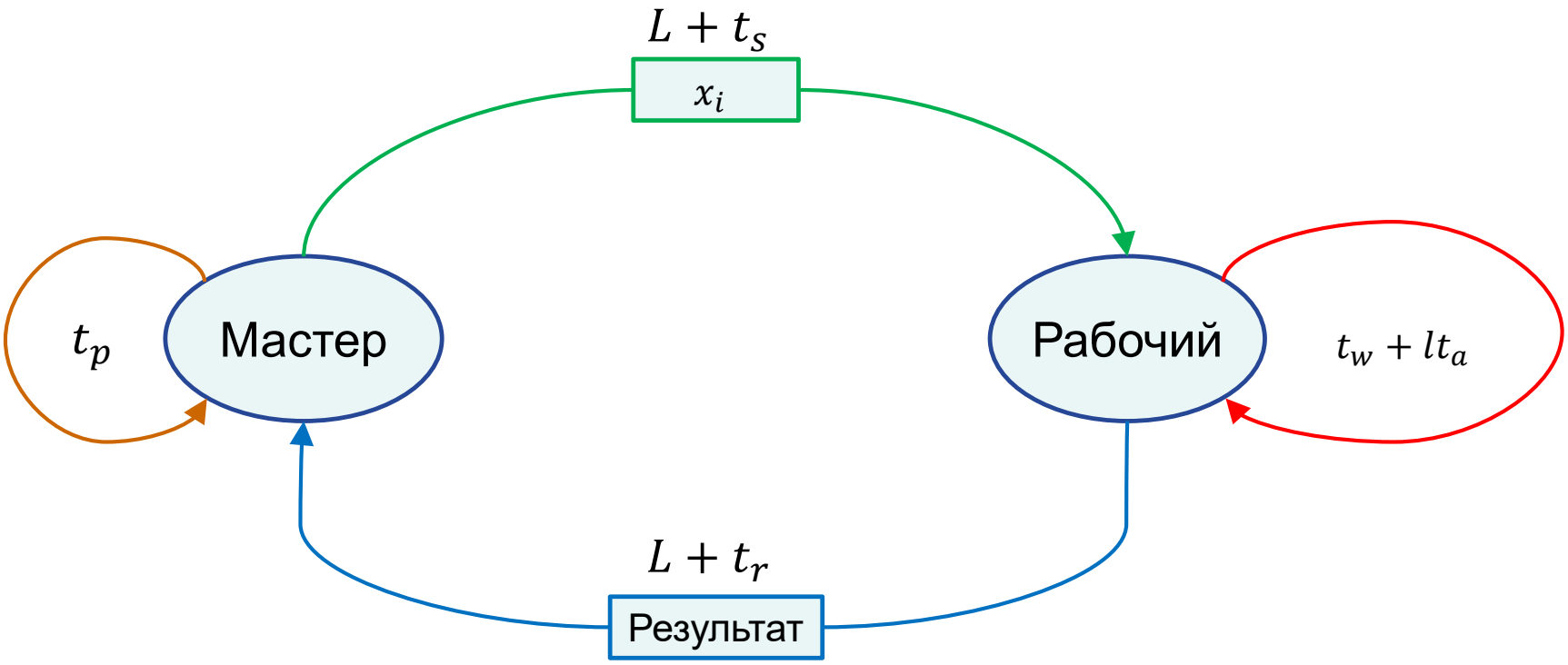
t_p – время, затрачиваемое мастером на вычисление следующего приближения и проверку условия завершения

Ускорение в модели BSF-MR



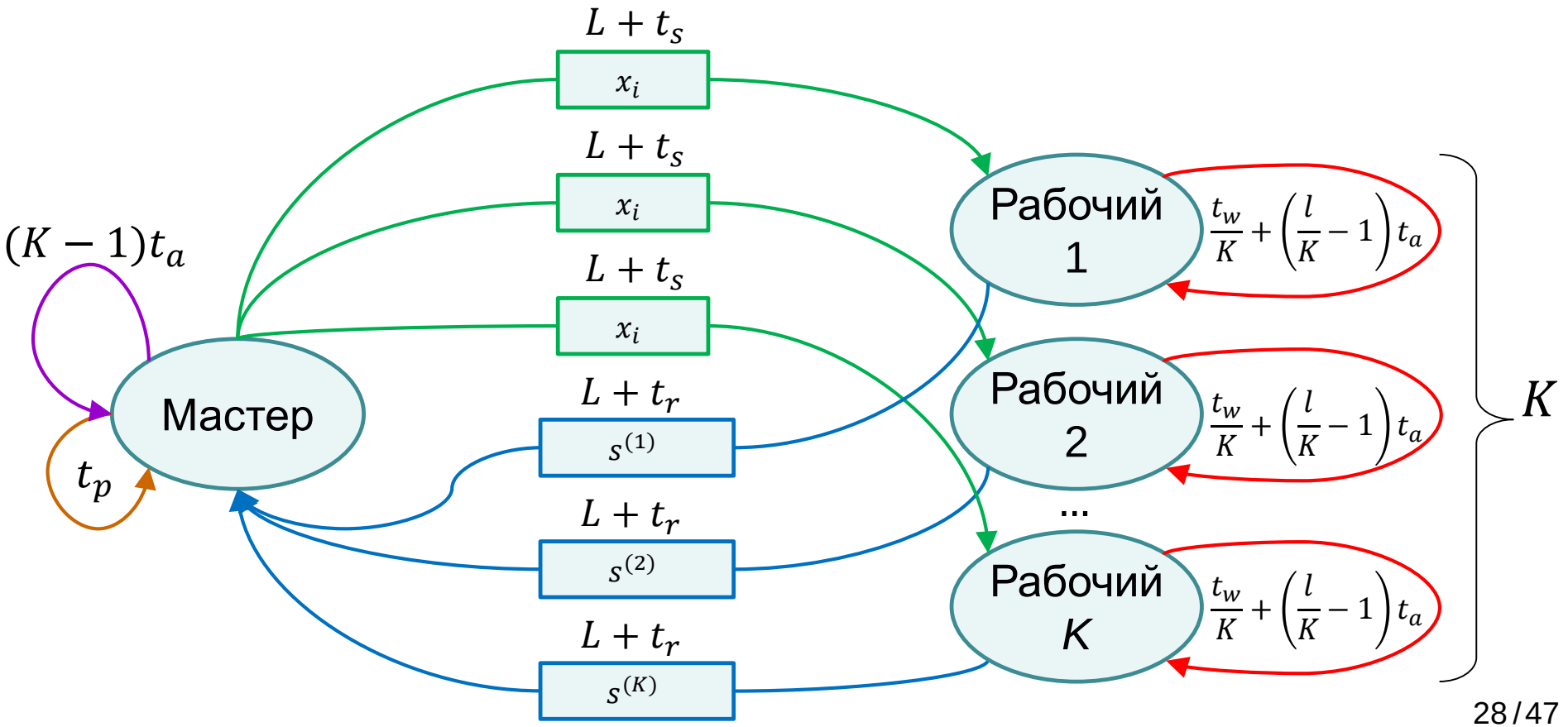
Время решения задачи системой из мастера и одного рабочего для BSF-MR

$$T_1 = L + t_s + t_w + lt_a + L + t_r + t_p$$



Время решения задачи системой из одного мастера и K рабочих для BSF-MR

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

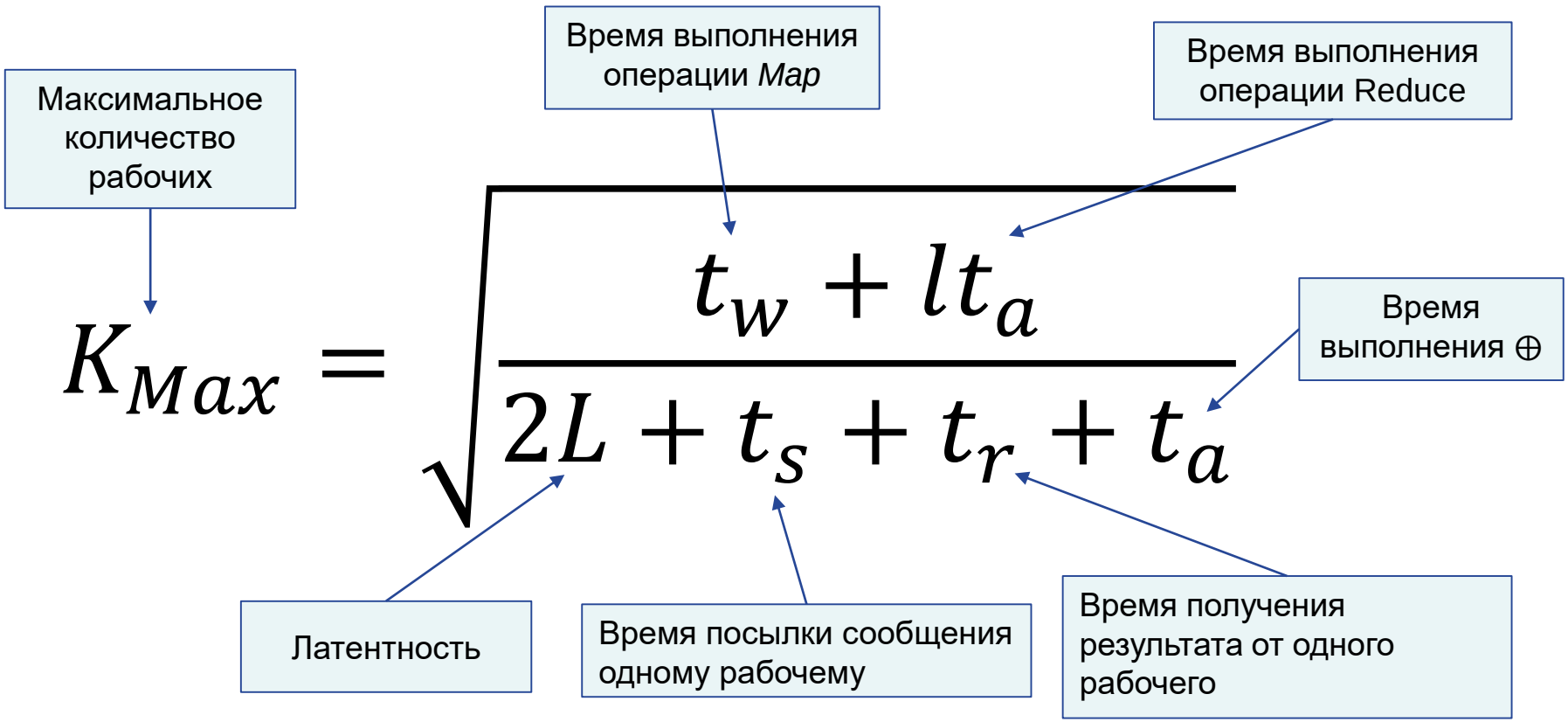


Ускорение для BSF-MR

$$a(K) = \frac{T_1}{T_K}$$

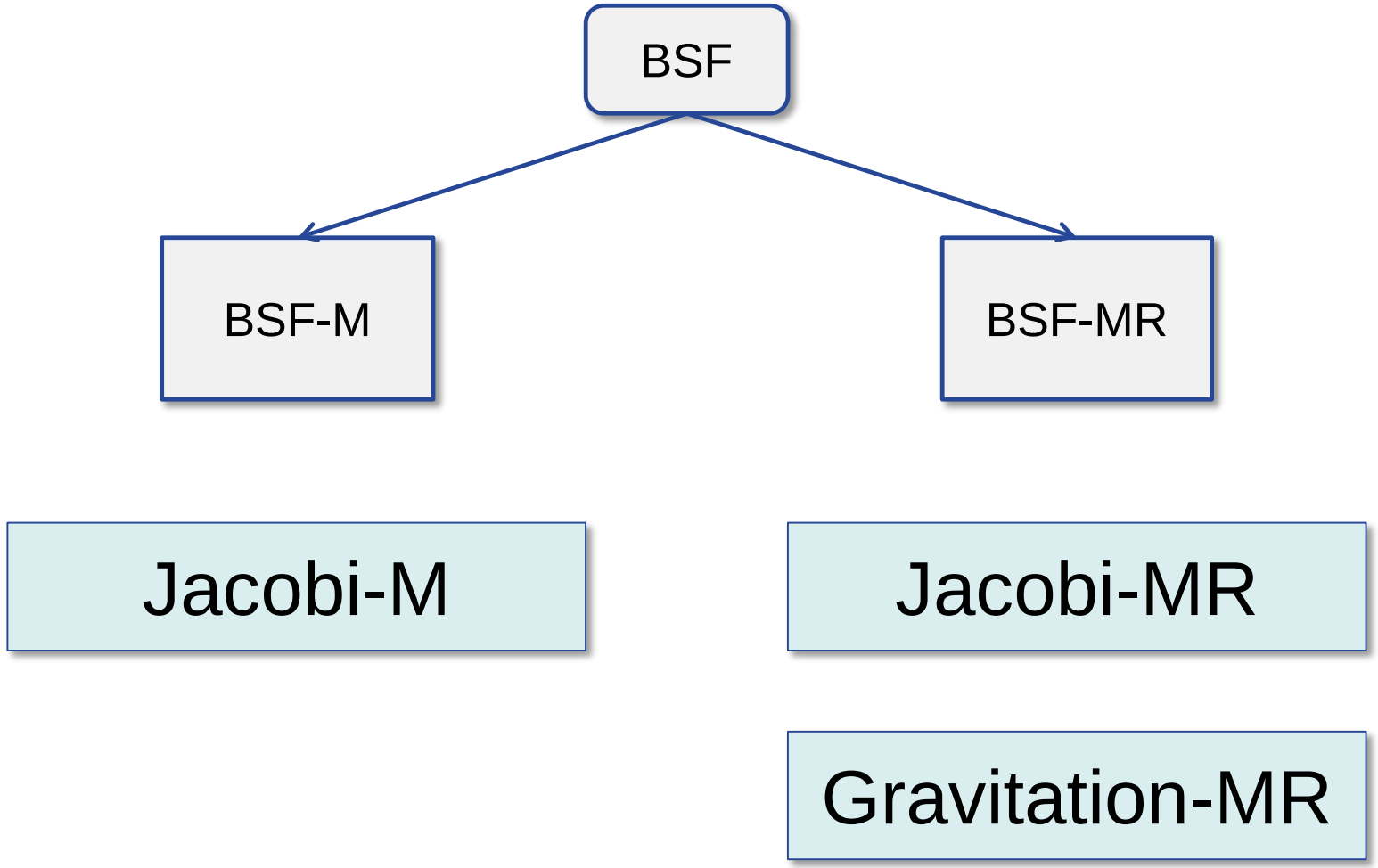
$$a_{BSF-MR}(K) = \frac{2L + t_s + t_r + t_p + t_w + lt_a}{K(2L + t_s + t_r + t_a) + (t_w + lt_a)/K - t_a + t_p}$$

Граница масштабируемости для BSF-MR



$$K_{Max} = \sqrt{\frac{t_w + lt_a}{2L + t_s + t_r + t_a}}$$

Примеры применения моделей



Алгоритм Jacobi для приближенного решения СЛАУ

$$Ax = b;$$

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

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

$$d = (d_1, \ldots, d_n); d_i = b_i \big/ a_{ii}.$$

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

Алгоритм Jacobi-M над списками

$$F_{x^{(k)}} = d_i + \sum_{j=1}^{n} c_{ij} x_i$$

1. $Input(A, b); Comput(C, d); k := 0; x^{(0)} := d$
2. $C_j^{row} := [c_{1j}, \dots, c_{nj}];\ x^{(k+1)} := Map(F_{x^{(k)}}, C_j^{row})$
3. Если $\left\| x^{(k+1)} - x^{(k)} \right\|^2 < \varepsilon$; перейти на шаг 5
4. $k := k + 1$; перейти на шаг 2
5. Стоп

BSF-оценки для алгоритма Jacobi-M для многопроцессорных систем с распределенной памятью

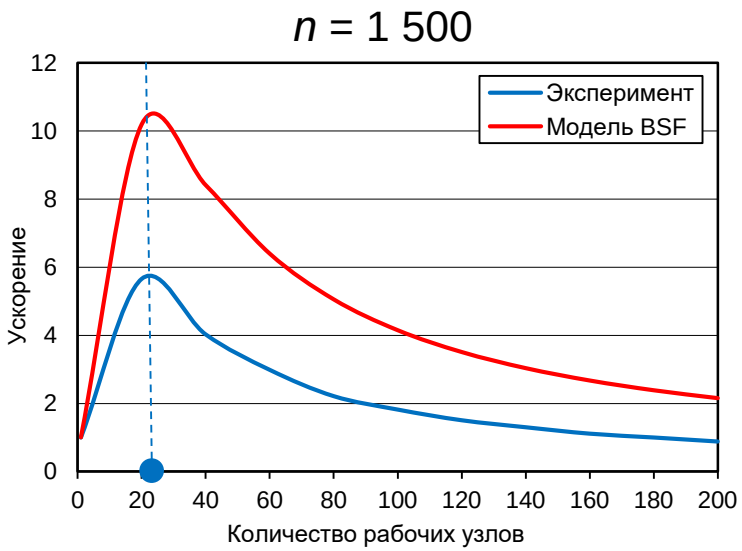
τ_{op} - время выполнения одной операции с плавающей точкой
τ_{tr} - время пересылки вещественного числа (без учета латентности)
n - количество уравнений
L - латентность

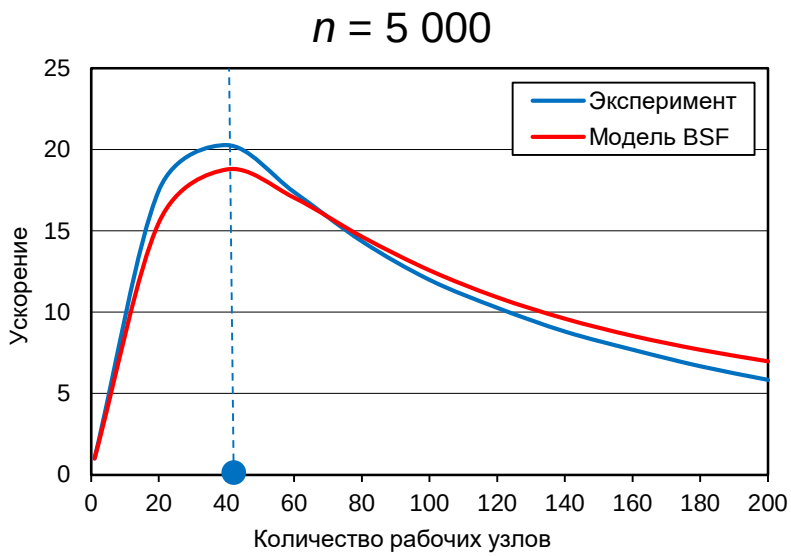
$$a_{Jacobi-M}(K) = \\ = \frac{K\left(2L + 2\tau_{tr}n + \tau_{op} \cdot (2n+2) + \tau_{op}2n^2\right)}{K^2(2L + \tau_{tr}n) + K\left(\tau_{tr}n + \tau_{op} \cdot (2n+2)\right) + \tau_{op}2n^2}$$

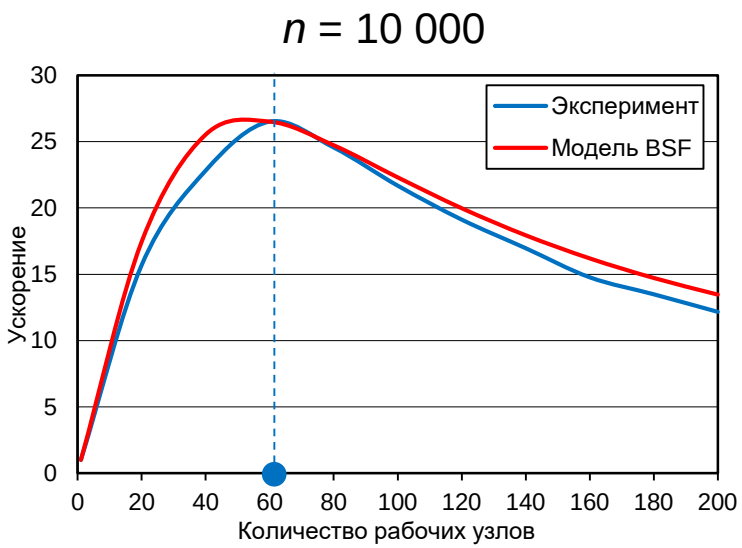
$$K^{Max}_{Jacobi-M} = \sqrt{\frac{\tau_{op}2n^2}{2L + \tau_{tr}n}}$$

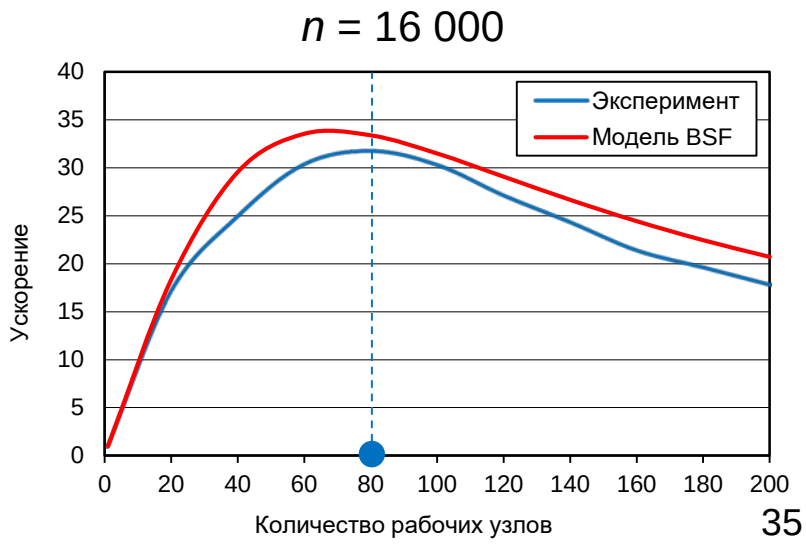
$$\boldsymbol{K_{max} = O\left(\sqrt{n}\right)}$$

Ускорение алгоритма Jacobi-M: теория и практика









Алгоритм Jacobi-MR над списками

$$F_x(j) = \left(c_{1j}x_j, \dots, c_{nj}x_j\right)$$

1. $Input(A, b); Comput(C, d); k := 0; x^{(0)} := d$
2. $C_i^{col} := Map\left(F_{x^{(k)}}, \left[c_{1j}, \dots, c_{nj}\right]\right)$
3. $x^{(k+1)} := Reduce\left(\oplus, C_i^{col}\right)$
4. $x^{(k+1)} := x^{(k+1)} + d$
5. Если $\left\|x^{(k+1)} - x^{(k)}\right\|^2 < \varepsilon$, перейти на шаг 7
6. $k := k + 1$; перейти на шаг 2
7. Стоп

BSF-оценки для алгоритма Jacobi-MR для многопроцессорных систем с распределенной памятью

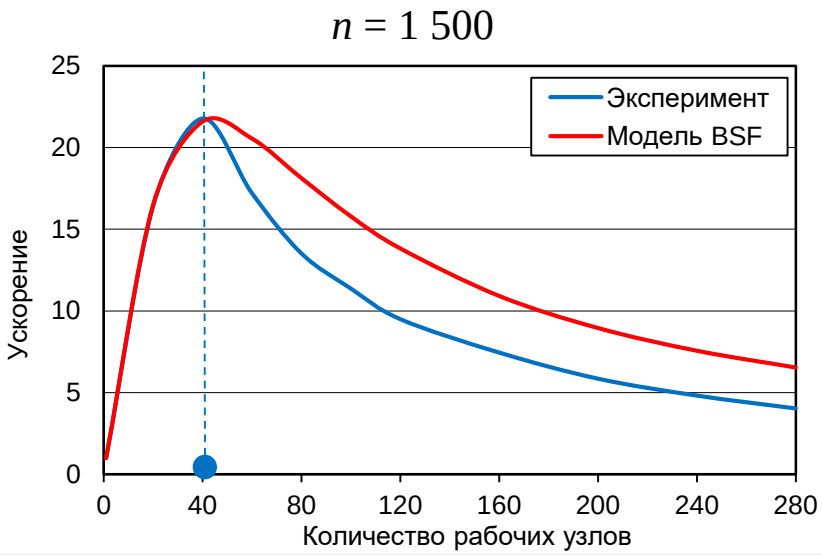
τ_{op} - время выполнения одной операции с плавающей точкой
τ_{tr} - время пересылки вещественного числа (без учета латентности)
n - количество уравнений
L - латентность

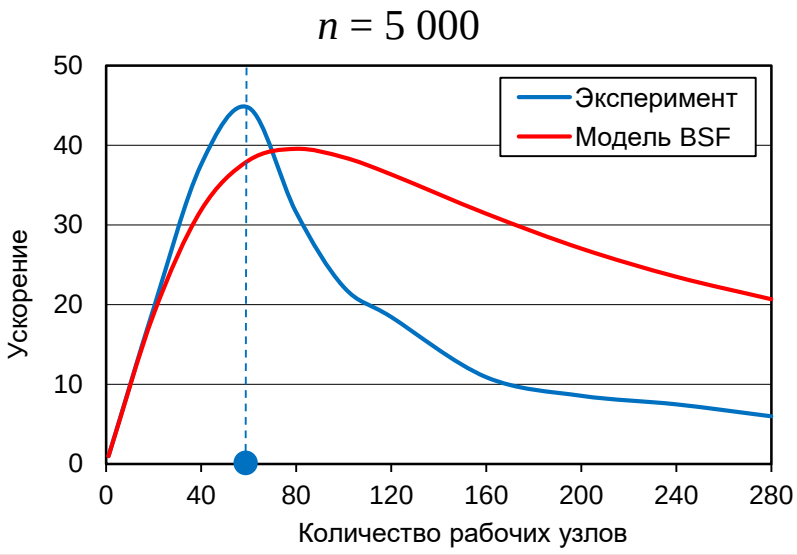
$$a_{Jacobi-MR}(K) = \frac{2(L + \tau_{tr}n) + \tau_{op}n(3n - K + 5)}{K^2\left(2\left(L + \tau_{op}n\right) + 3\tau_{op}n(n + K)\right)}$$

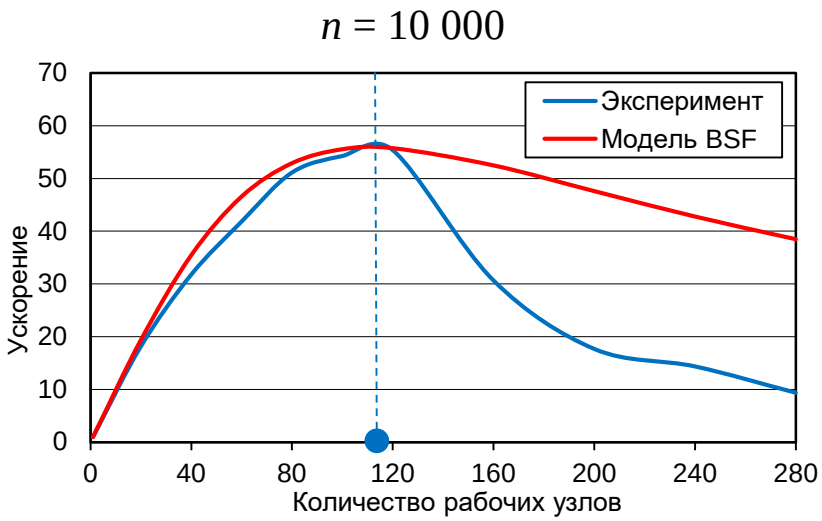
$$K_{Max} = \sqrt{\frac{\tau_{op}n(3n - K)}{2(L + \tau_{tr}n) + \tau_{op}n}}$$

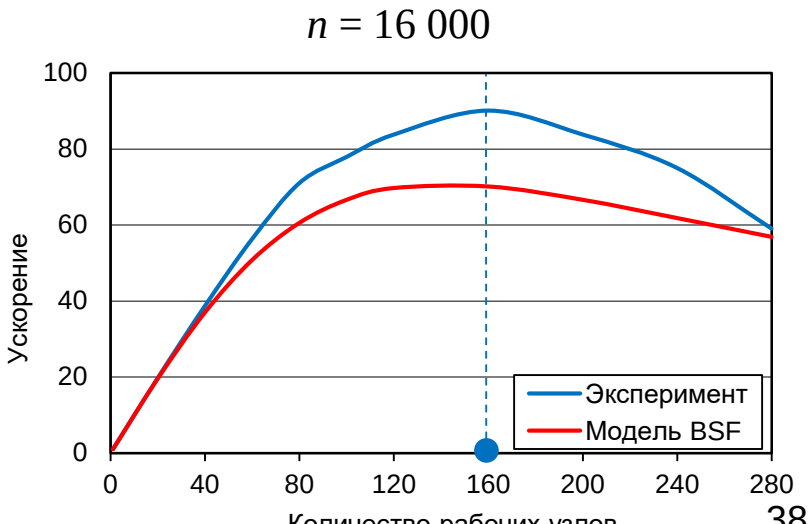
$$\boldsymbol{K_{max} = O\left(\sqrt{n}\right)}$$

Ускорение алгоритма Jacobi-MR: теория и практика

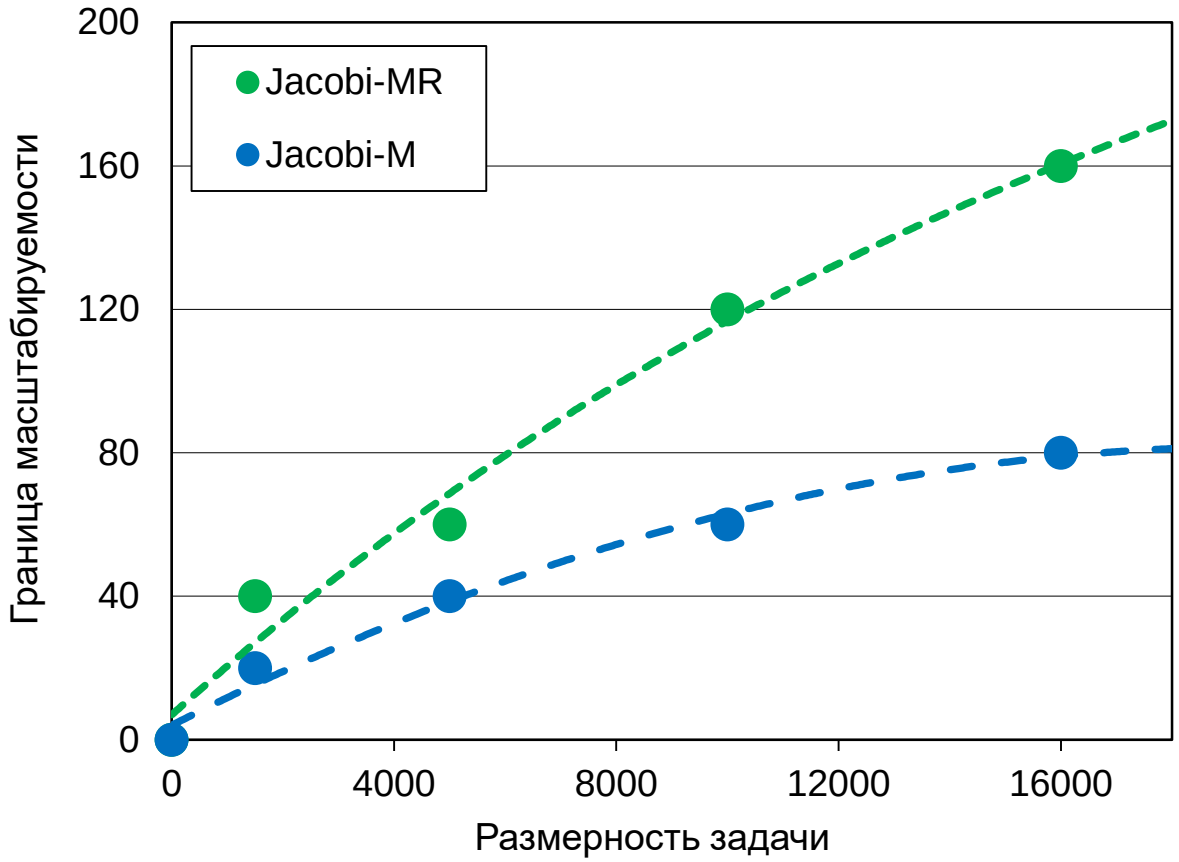




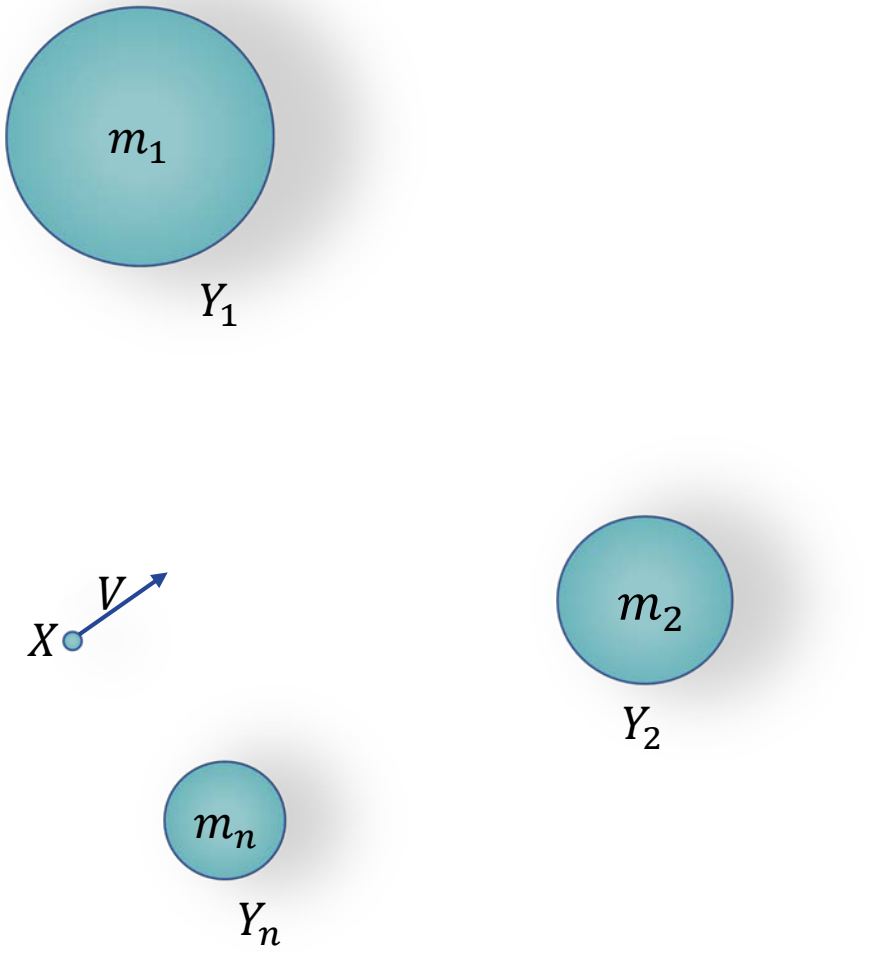




Зависимость границы масштабируемости алгоритмов Jacobi-MR и Jacobi-M от размерности задачи

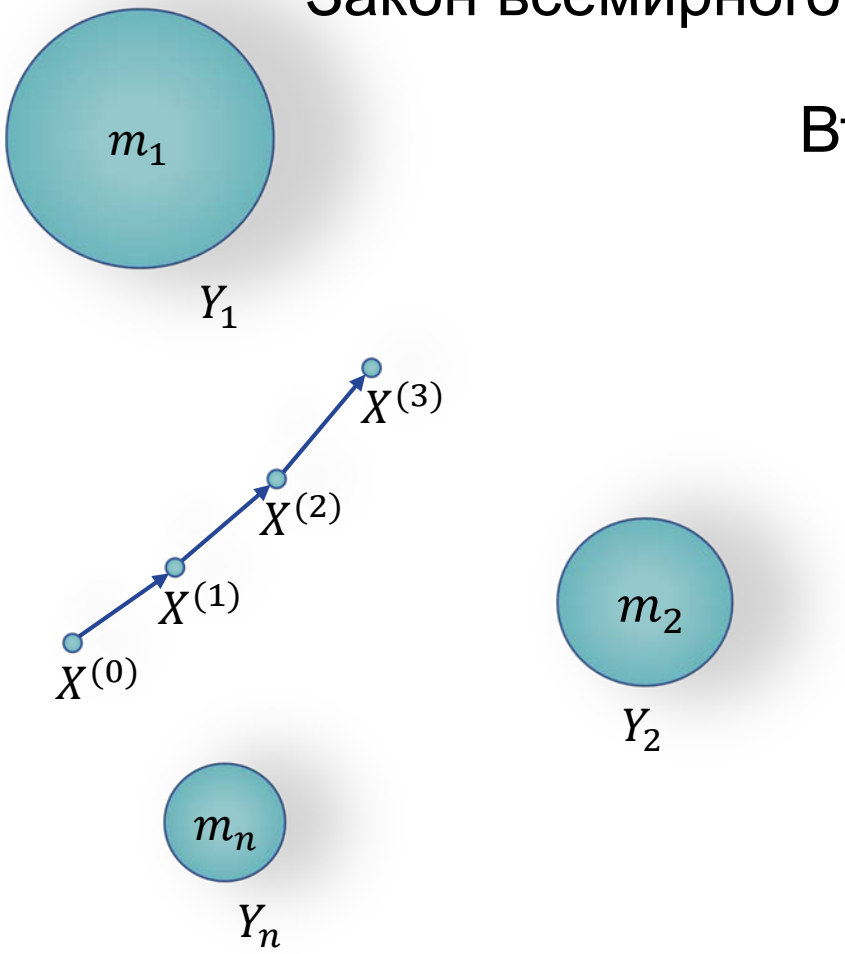


Гравитационная задача



Материальная точка X массой m_x движется с начальной скоростью V среди n неподвижных тел $Y_1, Y_2, \ldots, Y_n$ с большими массами $m_1, m_2, \ldots, m_n$.

Необходимо рассчитать траекторию точки X с шагом Δt



Закон всемирного тяготения: $F_i = G\frac{m_i m_x}{\|Y_i - X\|^3}(Y_i - X)$

Второй закону Ньютона: $A_i = \frac{F_i}{m_x}$

$$\alpha_X(Y_i) = G\frac{m_i}{\|Y_i - X\|^3}(Y_i - X)$$

$$A_i = \alpha_X(Y_i)$$

$$A = \sum_{i=1}^{n} A_i$$

$$V^{(k+1)} = V^{(k)} + A \cdot \Delta t$$

$$X^{(k+1)} = X^{(k)} + V^{(k+1)} \cdot \Delta t$$

Алгоритм Gravitation-MR (K – количество итераций)

1. $k := 0; x := 0; V^{(0)} := 0;\ Input([Y_1, \dots, Y_n])$
2. $[A_1, \dots, A_n] := Map(\alpha_X, [Y_1, \dots, Y_n])$
3. $A := Reduce(+, [A_1, \dots, A_n])$
4. $V^{(k+1)} := V^{(k)} + A \cdot \Delta t$
5. $X^{(k+1)} := X^{(k)} + V^{(k+1)} \Delta t$
6. Если $k \geq K$, перейти на шаг 8
7. $k := k + 1$; перейти на шаг 2
8. Стоп

BSF-оценки для алгоритма Gravitation-MR для многопроцессорных систем с распределенной памятью

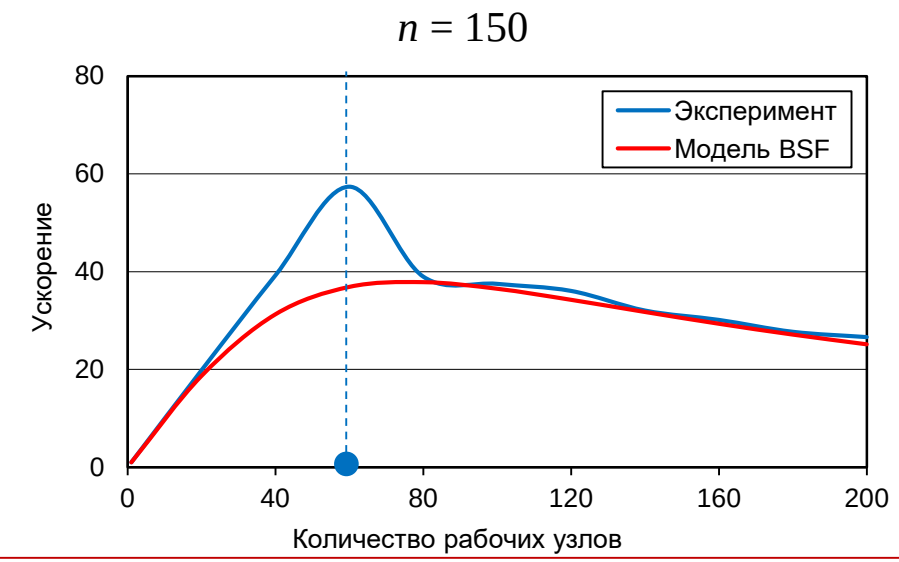
τ_{op} - время выполнения одной операции с плавающей точкой
τ_{tr} - время пересылки вещественного числа (без учета латентности)
L - латентность
n - количество тел

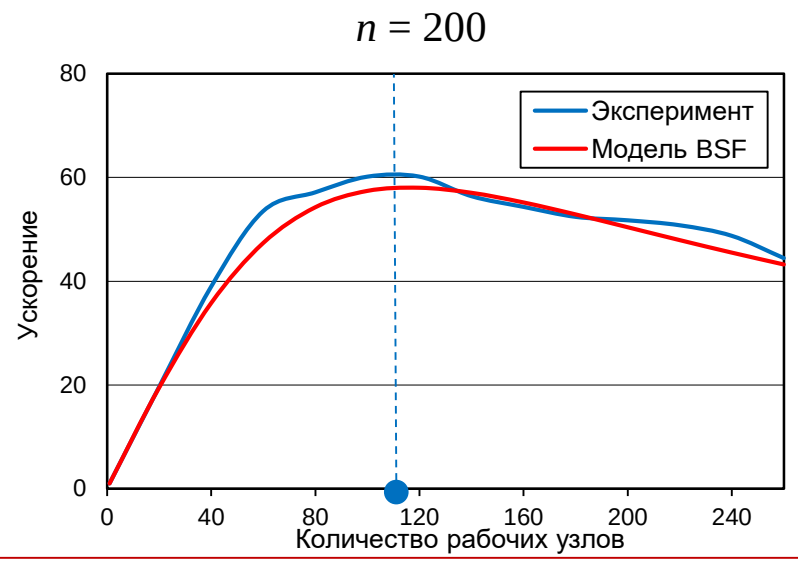
$$a_{BSF-MR}(K) = \frac{2L + 9\tau_{tr} + 13\tau_{op} + \tau_{op}\left(11n + \frac{n-K}{K}\right) + 3\tau_{op} \cdot l}{K\left(2L + 9\tau_{tr} + 13\tau_{op}\right) + \left(\tau_{op}\left(11n + \frac{n-K}{K}\right) + 3\tau_{op} \cdot l\right)/K + 10\tau_{op}}$$

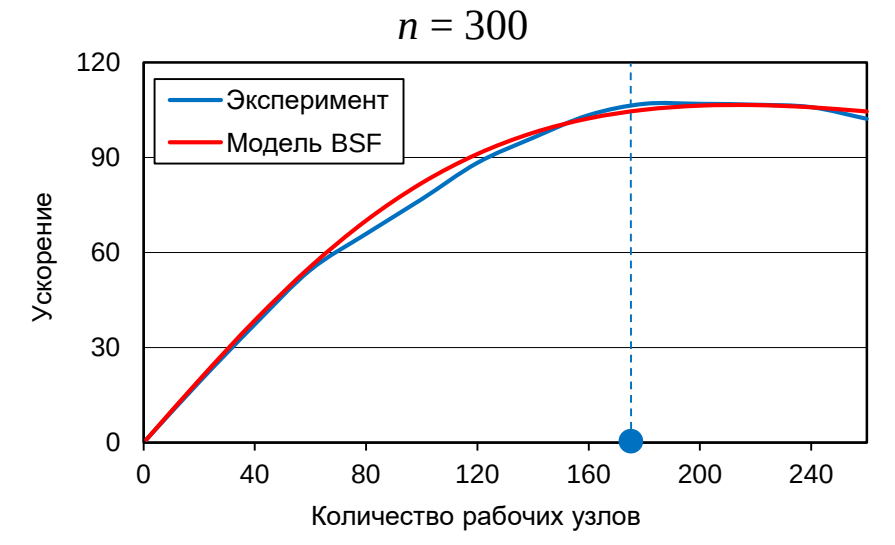
$$K_{Max} = \sqrt{\frac{\tau_{op}\left(11n + \frac{n-K}{K}\right) + 3\tau_{op} \cdot l}{2L + 9\tau_{tr} + 13\tau_{op}}}$$

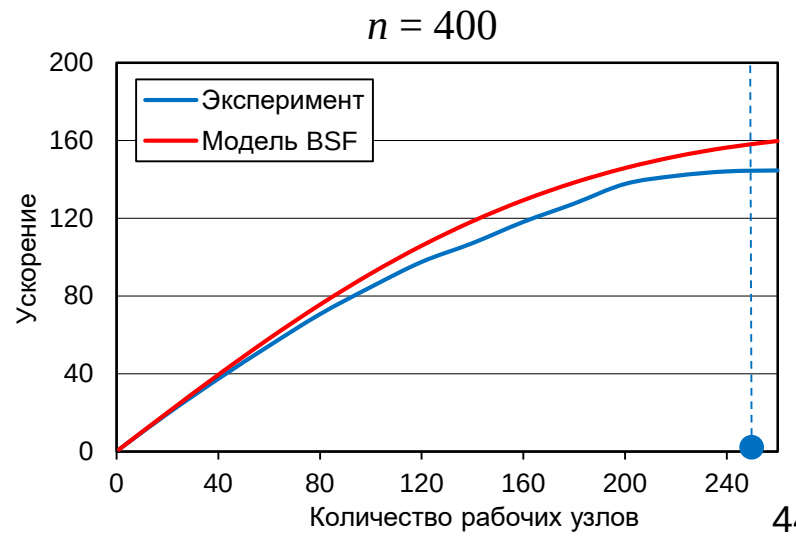
$$\boldsymbol{K_{max} = O(n)}$$

Ускорение алгоритма Gravitation-MR: теория и практика

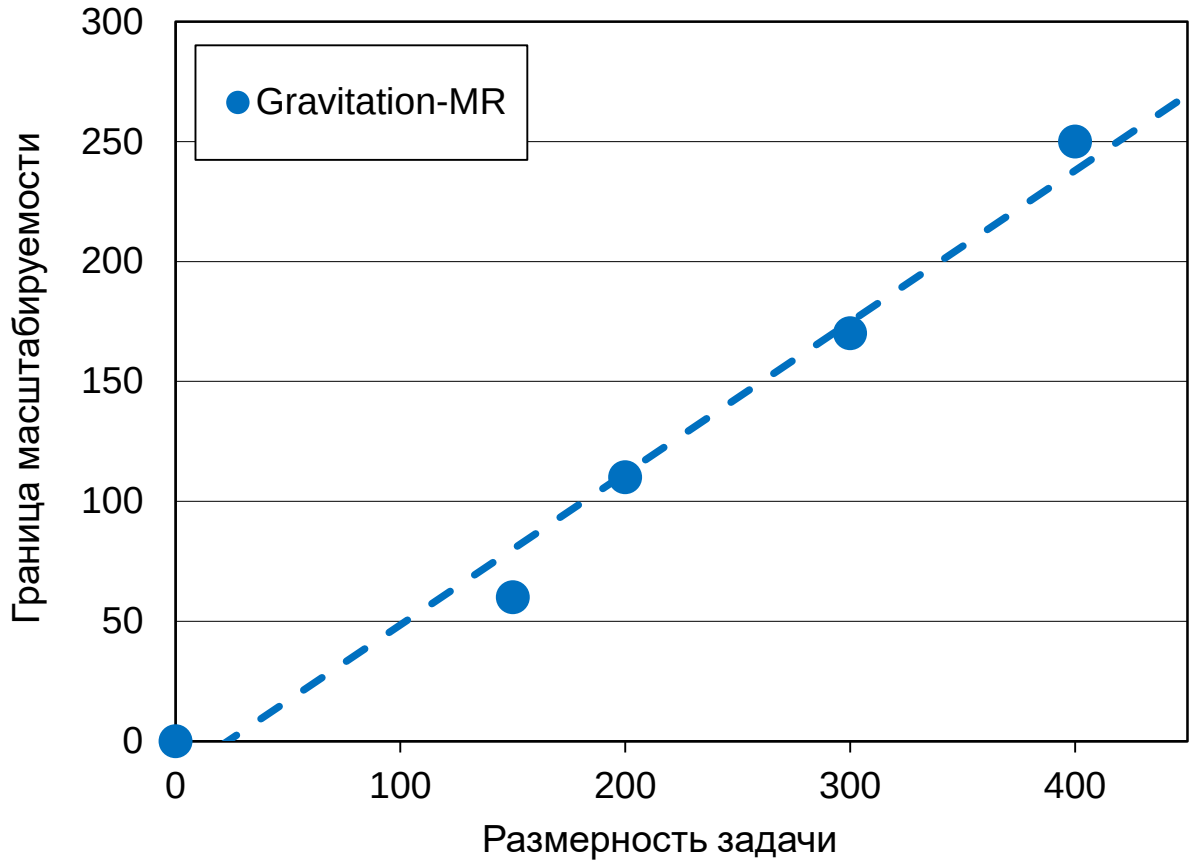








Зависимость границы масштабируемости алгоритма Gravitation-MR от размерности задачи



Исходные коды

https://github.com/nadezhda-ezhova/Gravitation-MR
https://github.com/nadezhda-ezhova/Jacobi-MR
https://github.com/nadezhda-ezhova/Jacobi-M

Спасибо за внимание!