Международная конференция «Суперкомпьютерные дни в России» (Москва, 24 - 25 сентября 2018 г.)

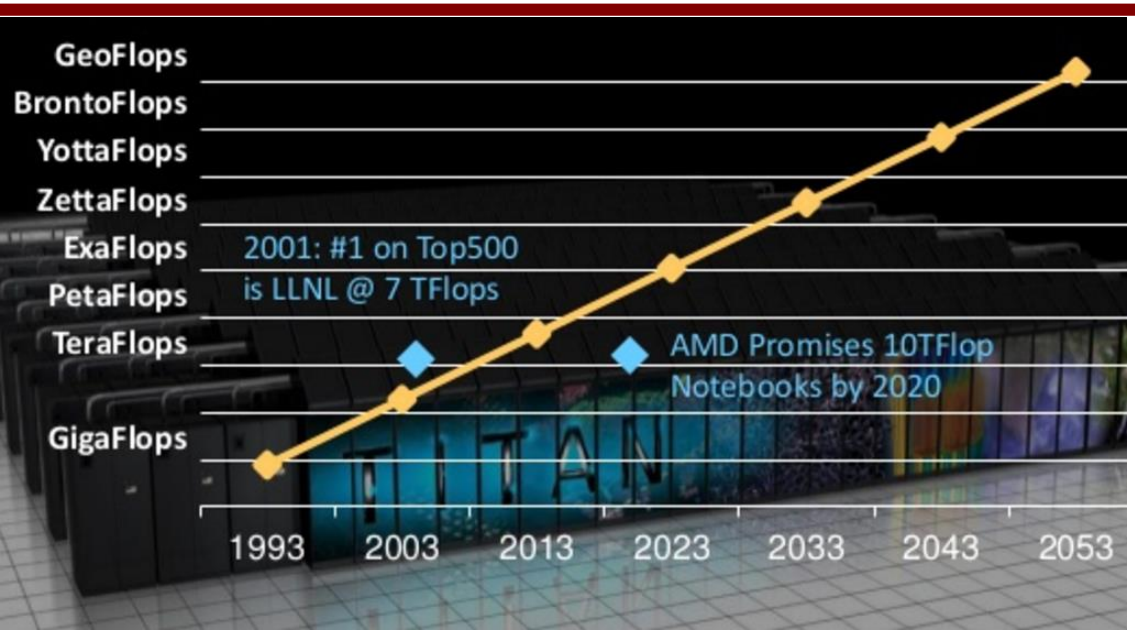
Исследование масштабируемости модифицированного алгоритма Чиммино для линейных неравенств

д.ф.-м.н., Л.Б. Соколинский,
к.ф.-м.н., И.М. Соколинская

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).

Экзафлопный вызов



- Наличие сверхбольших задач, сводящихся к решению систем линейных неравенств или уравнений
- Численные параллельные алгоритмы, разработанные для общей памяти, не масштабируются на многопроцессорных системах с распределенной памятью:
  - Хороший параллельный алгоритм для общей памяти плохо масштабируется на распределенной памяти
  - Хорошо масштабируемый распределенный алгоритм не эффективен на общей памяти

Необходимо создавать библиотеки параллельных численных алгоритмов нового класса, ориентированные на экзафлопные вычислители с распределенной памятью

Ускорение – главная характеристика масштабируемого алгоритма

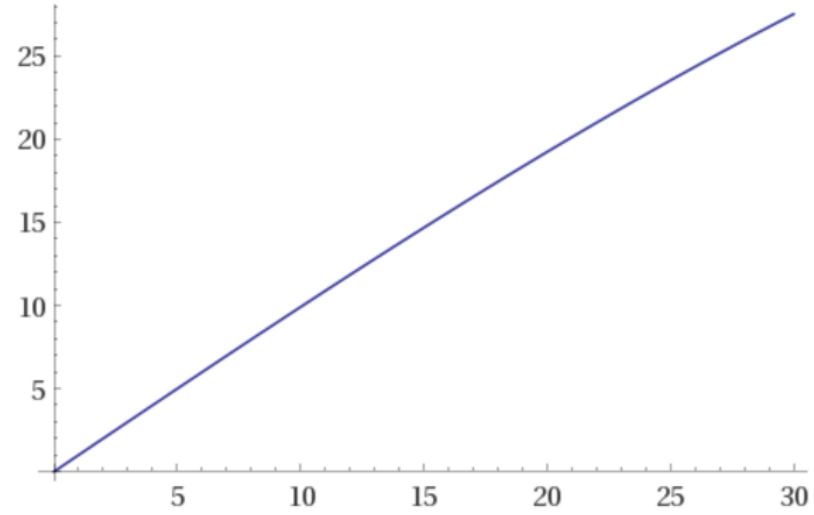
$$a(K) = \frac{t_1}{t_K}$$

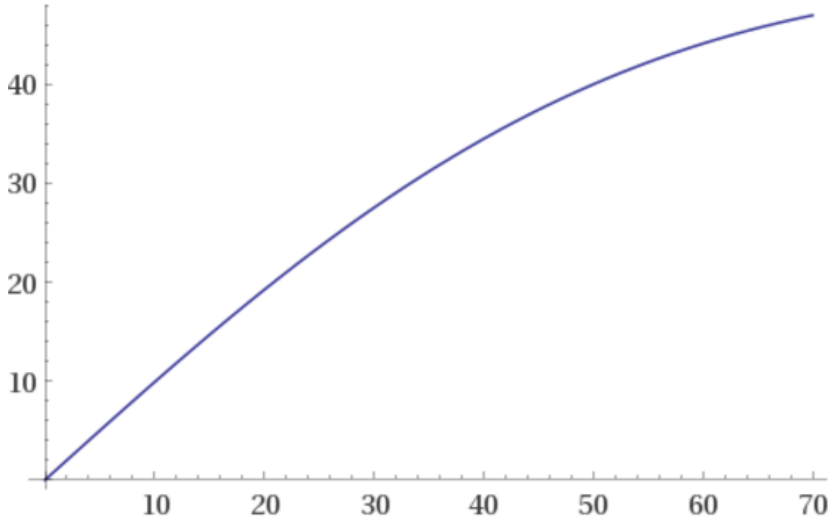
Время решения задачи на 1 узле

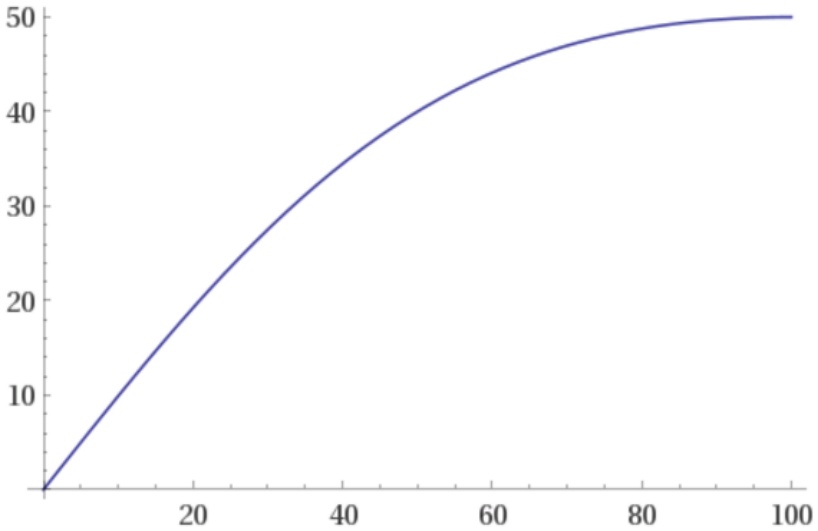
Время решения задачи на *K* узлах

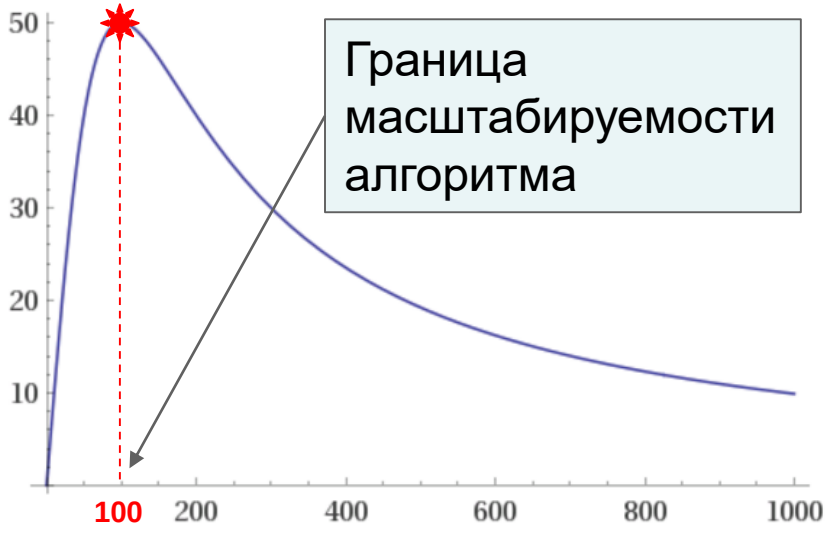
Количество процессорных узлов

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





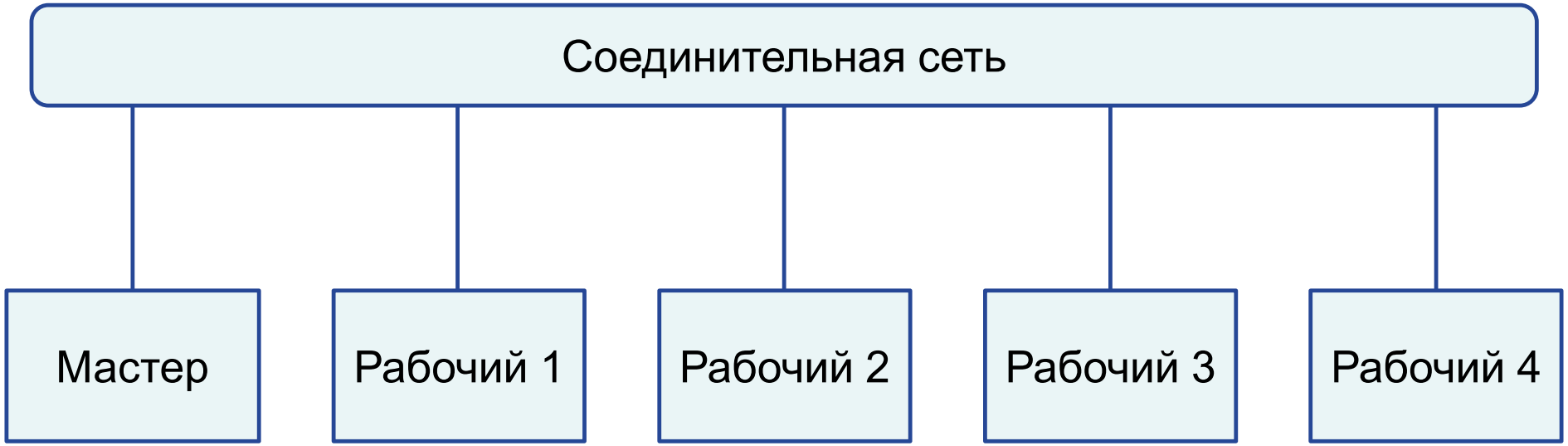




Модель параллельных вычислений BSF (Bulk-Synchronous Farm)

- *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

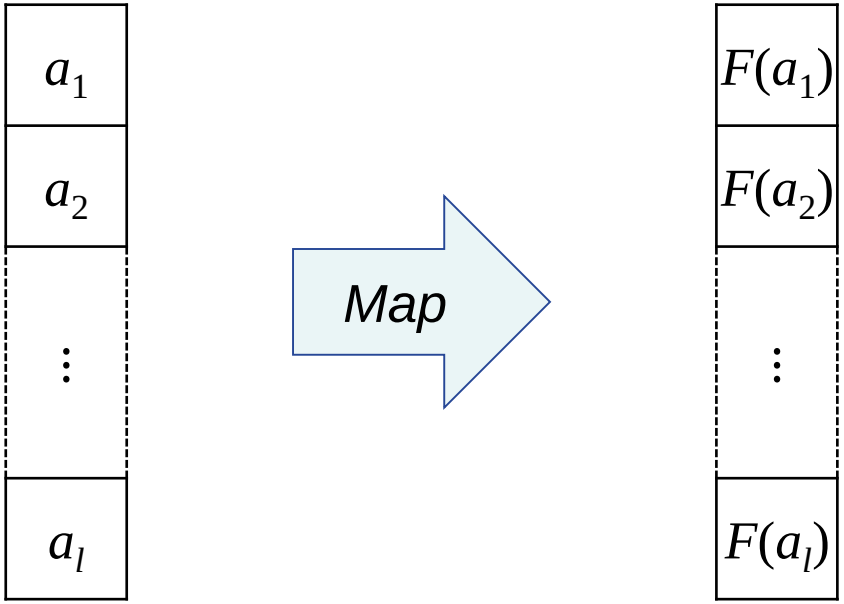


Процессорные узлы

Представление алгоритма в виде операций над списками

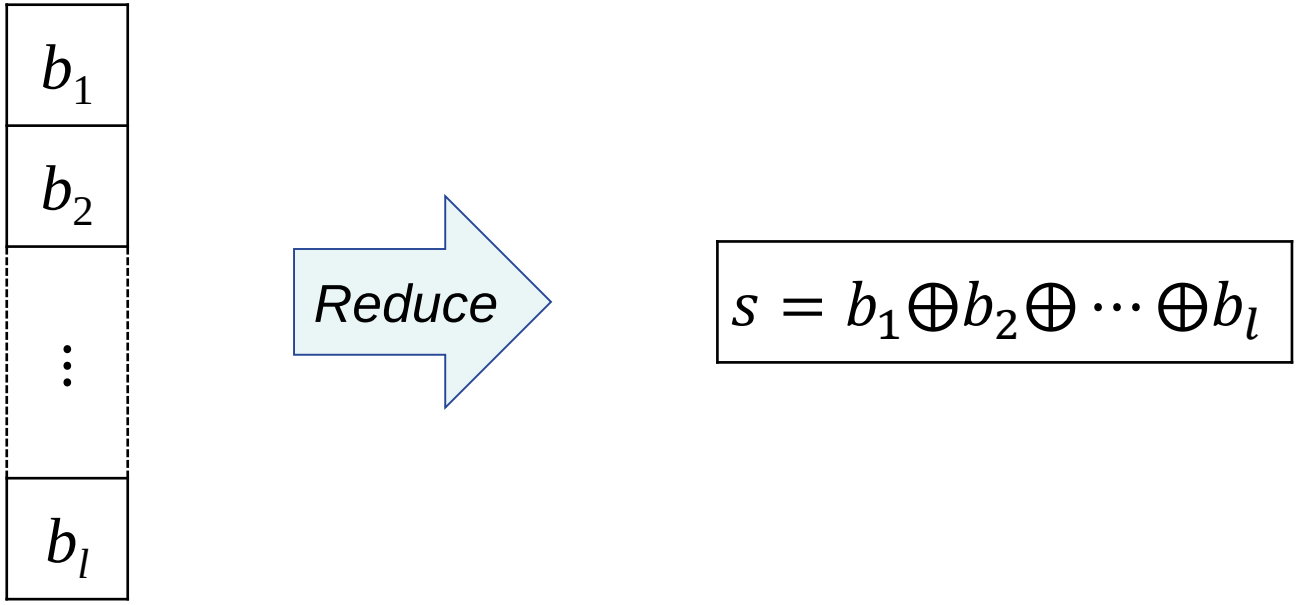
- Функции высшего порядка:
  - Map
  - Reduce

Функция высшего порядка *Map*



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

Функция высшего порядка *Reduce*



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

Шаблон итерационного BSF-алгоритма над списками

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 BSF-алгоритма над списками

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 модели BSF

K – количество рабочих

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)

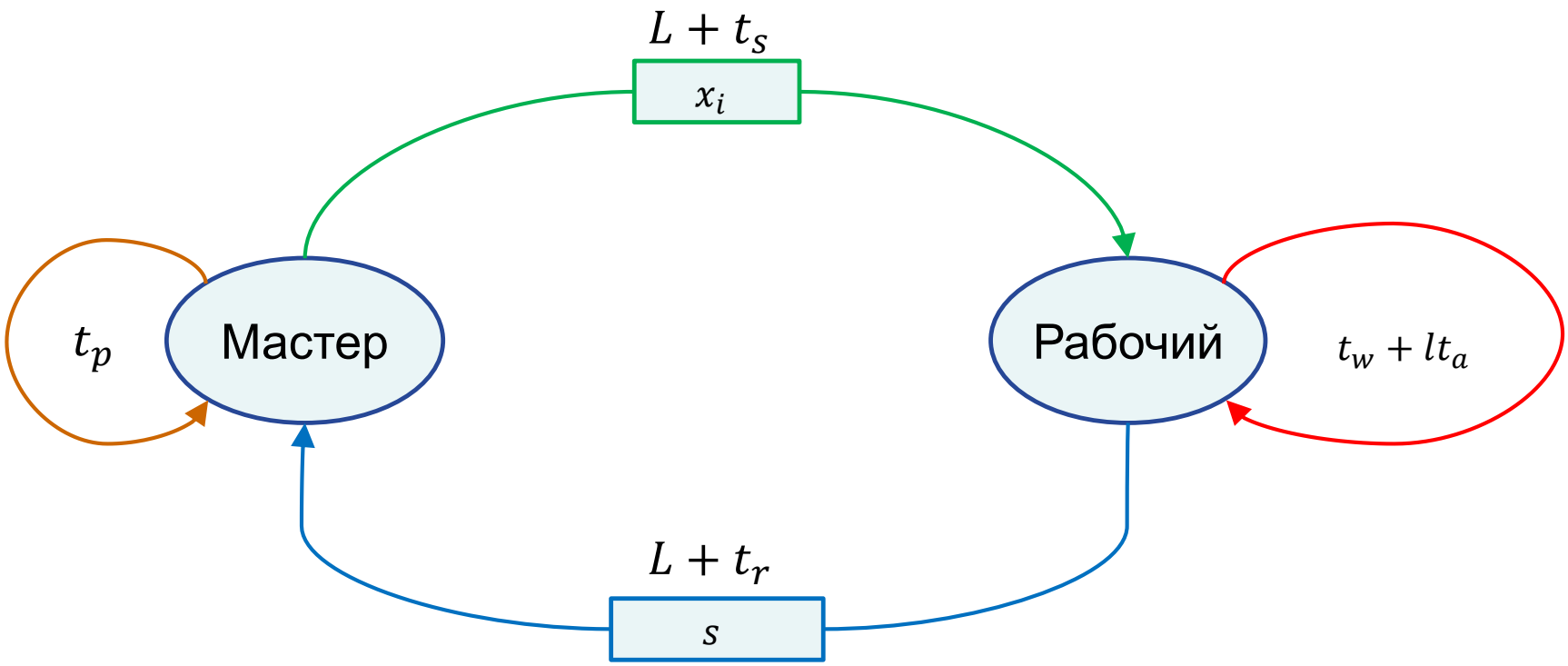
t_a – время, необходимое для выполнения одной операции $\oplus$

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

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

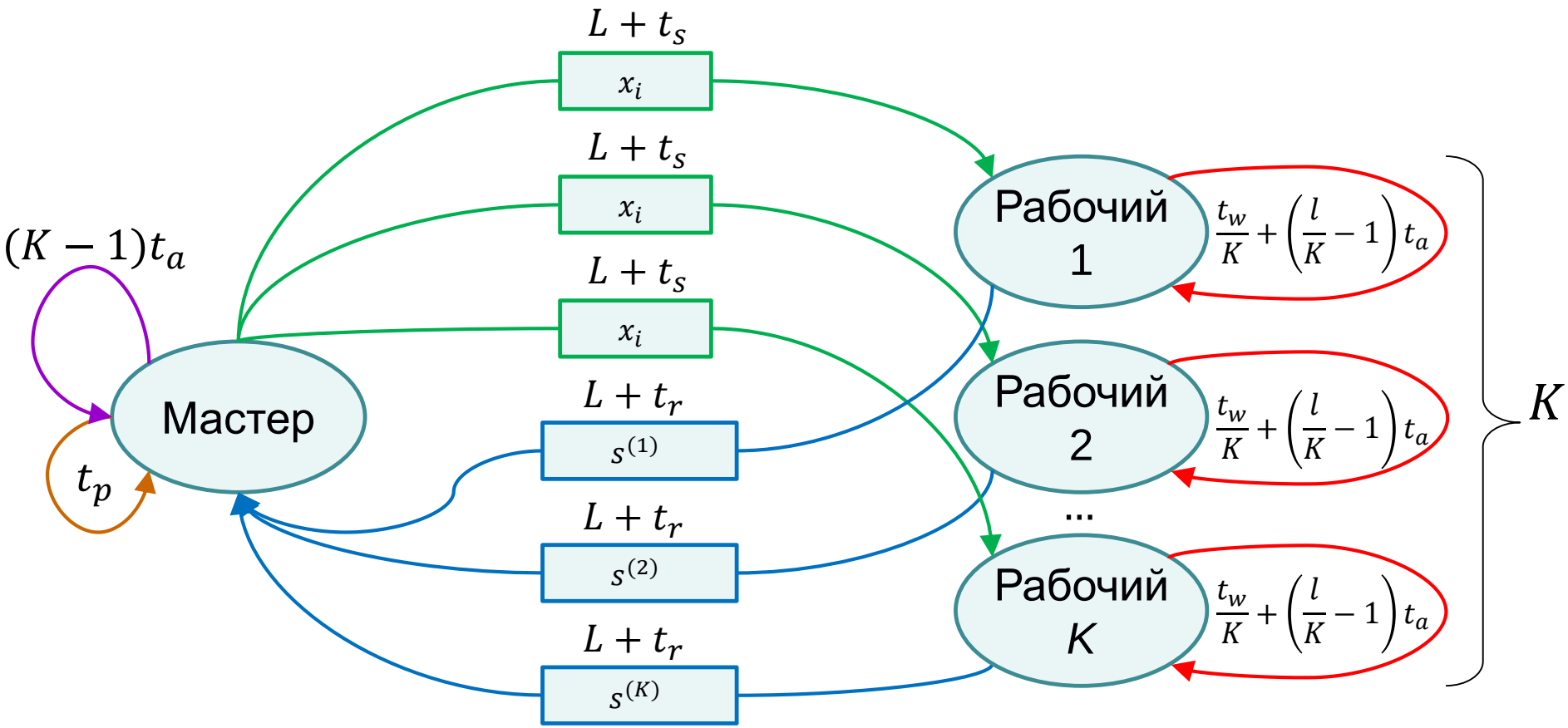
Время T_1 решения задачи системой из одного мастера и одного рабочего (сек.)

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



Время T_K решения задачи системой из одного мастера и K рабочих (сек.)

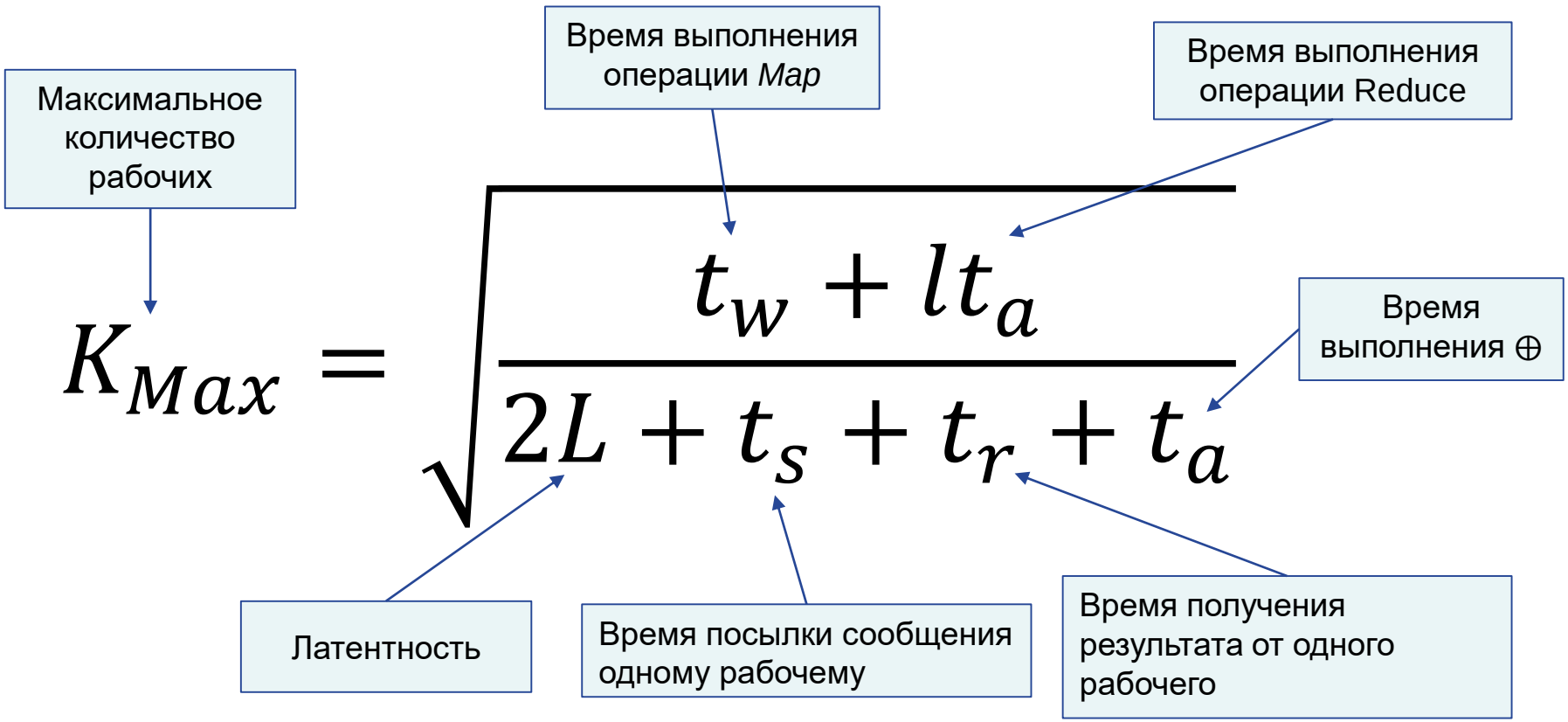
$$T_K = K(L + t_s) + \frac{t_w}{K} + \left(\frac{l}{K} - 1\right) t_a + K(L + t_r) + (K - 1)t_a + t_p$$



Ускорение BSF-алгоритма

$$a(K) = \frac{T_1}{T_K} = \frac{2L + t_s + t_r + t_p + t_w + lt_a}{K(2L + t_s + t_r + t_a) + (t_w + lt_a)/K - t_a + t_p}$$

Граница масштабируемости BSF-алгоритма



Алгоритм Чиммино для системы линейных неравенств (Gianfranco Cimmino, 1938)

$\langle a_i, x\rangle - b_i \le 0 \quad (i = 1, \dots, m)$ - система линейных неравенств

$H_i = \{x \in \mathbb{R}^n | \langle a_i, x\rangle = b_i\}$ - гиперплоскость, соответствующая i-тому неравенству

$\pi_{H_i}(x) = x + \frac{b_i - \langle a_i, x\rangle}{\|a_i\|^2} a_i$ - проекция x на гиперплоскость H_i

$\rho_{H_i}(x) = \pi_{H_i}(x) - x = \frac{b_i - \langle a_i, x\rangle}{\|a_i\|^2} a_i$ - ортогональное отражение x относительно H_i

$x_{k+1} = x_k + \frac{\lambda}{m} \sum_{i=1}^{m} \rho_{H_i}(x_k)$ - сходится к решению системы исходных неравенств при $0 < \lambda < 2$

BSF-представление алгоритма Чиммино

$$F_x(H_i) = \rho_{H_i}(x)$$

1. $i := 0; x := \mathbf{0}$
2. $i := i + 1$
3. $[x_i^1, \ldots, x_i^m] := Map\left(F_{x_i}, [H_1, \ldots, H_m]\right)$
4. $s_i := Reduce\left(+, [x_i^1, \ldots, x_i^m]\right)$
5. $x_{i+1} := x_i + \frac{\lambda}{m} s_i$
6. **if** $\|x_{i+1} - x_i\|^2 < \varepsilon$ **goto** 8
7. **goto** 2
8. **stop**

BSF-оценки алгоритма Чиммино для многопроцессорных систем с распределенной памятью

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

$$a(K) = \frac{T_1}{T_K} = \frac{2(L+\tau_{tr}n)+\tau_{op}\left(m(n^2+n+2)+4n\right)}{2(L+\tau_{tr}n)K+\tau_{op}\left(\frac{m(n^2+n+2)}{K}+Kn+3n\right)}$$

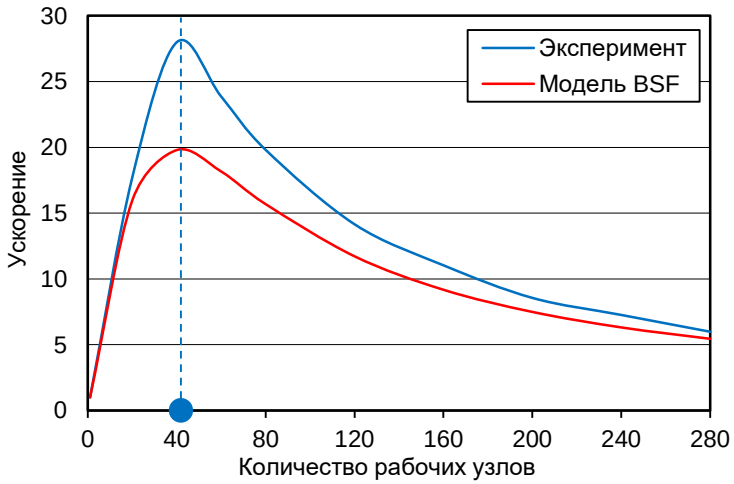
$$\boldsymbol{K_{max} = O(n)} \text{ (при } m \geq n)$$

BSF-реализация алгоритма Чиммино на C++ и MPI

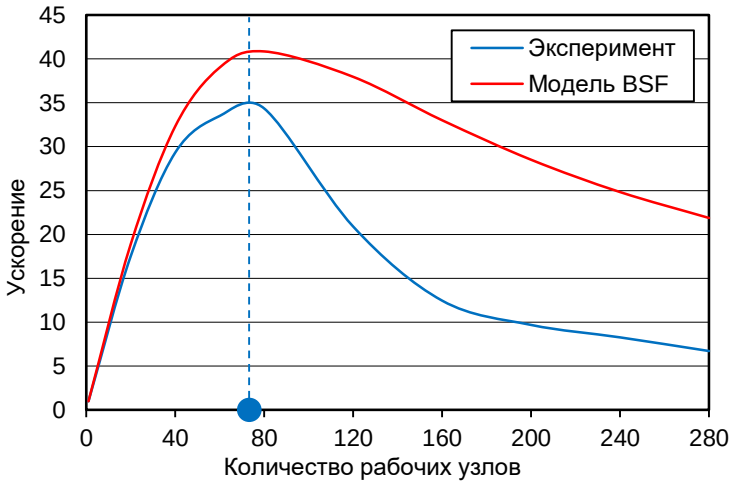
https://github.com/leonid-sokolinsky/BSF-Cimmino

Ускорение алгоритма Чиммино: теория и практика

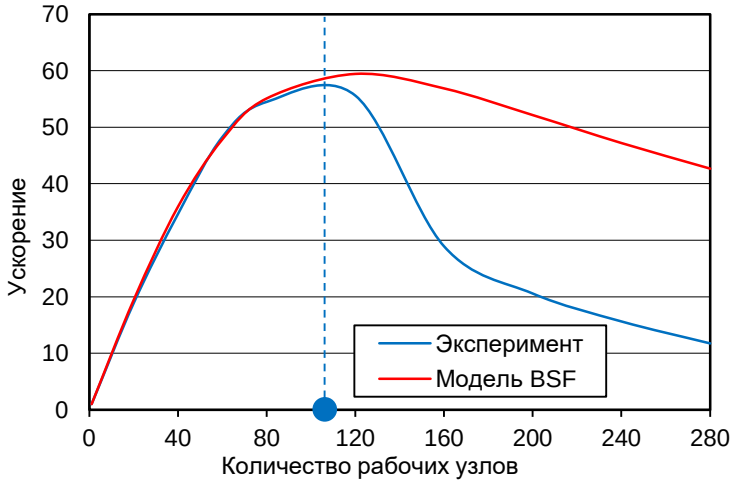
$n = 1\,500,\ m = 3\,002$



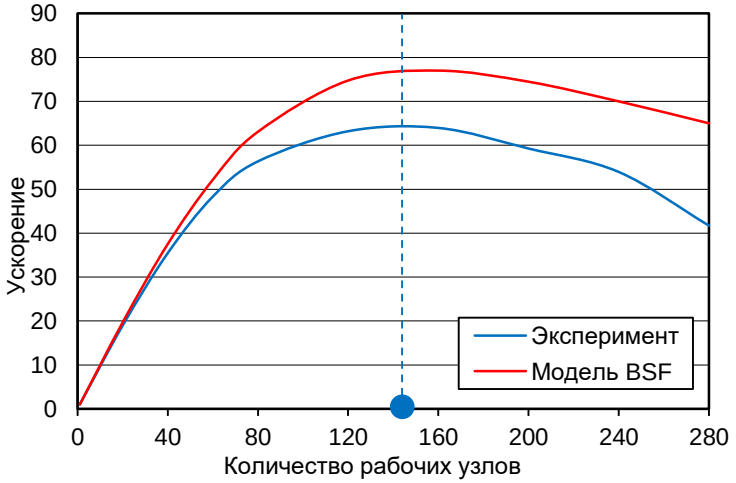
$n = 5\,000,\ m = 10\,002$



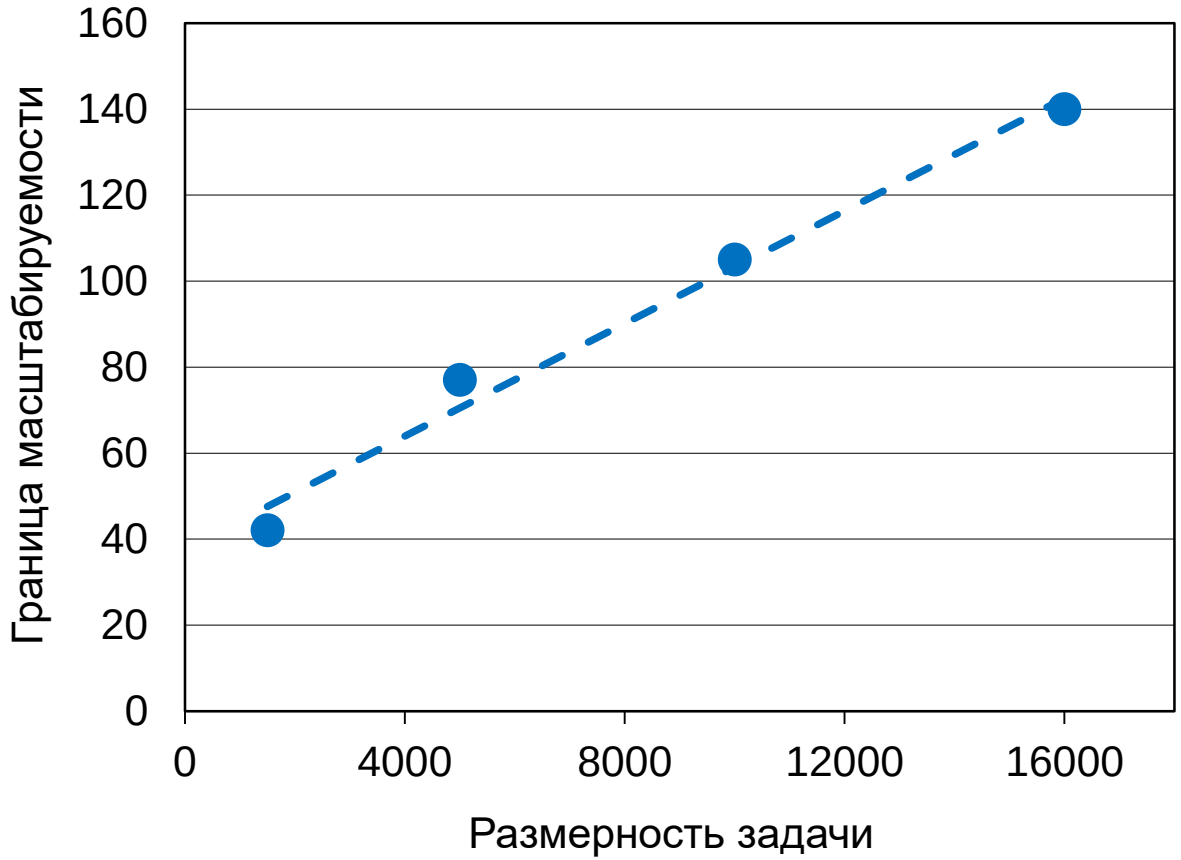
$n = 10\,000,\ m = 20\,002$



$n = 16\,000,\ m = 32\,002$



Зависимость границы масштабируемости алгоритма Чиммино от размерности задачи



Спасибо за внимание!