Параллельная матричная модификация метода муравьиных колоний для решения параметрических задач на гетерогенных вычислителях

Титов Юрий Павлович, МАИ

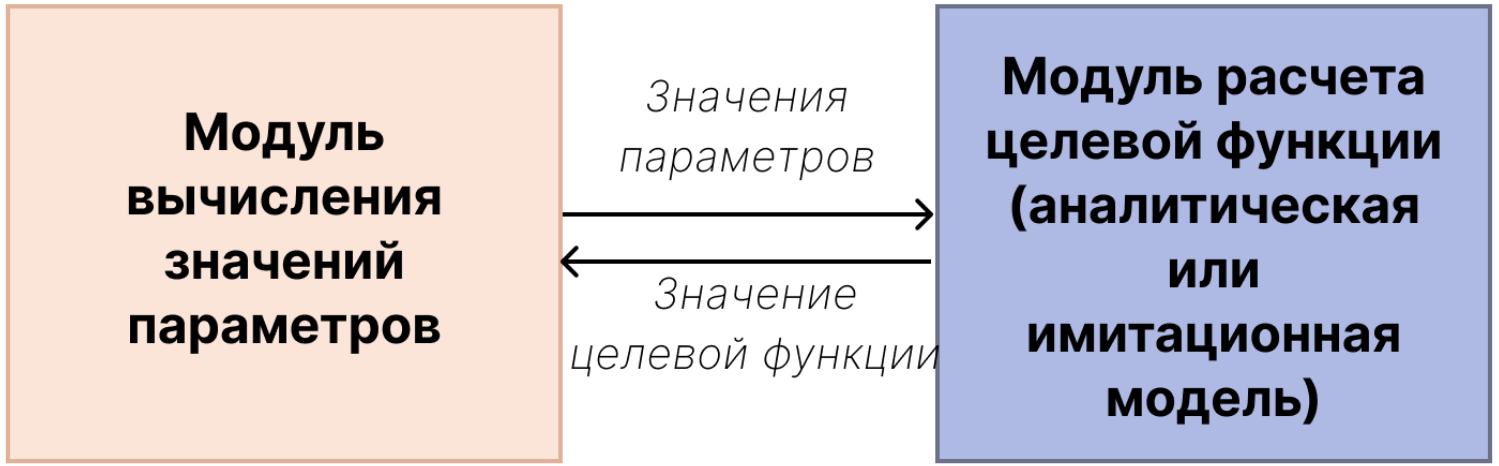
kalengul@mail.ru

Судаков Владимир Анатольевич,
ИПМ им. Келдыша

sudakov@ws-dss.com

Параметрическая задача. Постановка.

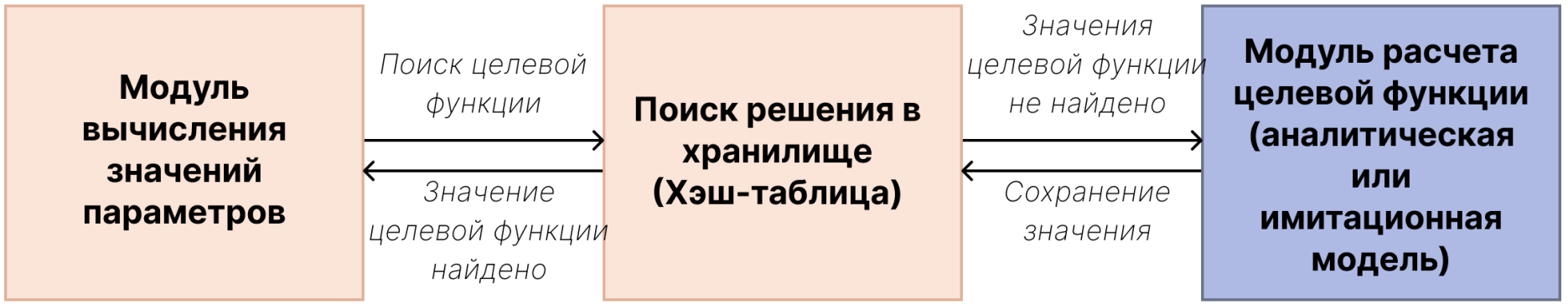
- Множество параметров: $\boldsymbol{P} = \{p_1, p_2 \dots p_i, \dots p_n\}$.
- Множество значений: $\boldsymbol{V_i} = \{v_{1,i}, v_{2,i} \dots v_{j,i}, \dots v_{m_i,i}\}, i = \overline{1,n}; |\boldsymbol{V_i}| = m_i > 0$
- Решение: $\boldsymbol{X_k} = (x_{1,k}, x_{2,k} \dots x_{i,k}, \dots x_{n,k}), |\boldsymbol{X_k}| = n, \forall i \exists j: (x_{i,k} = v_{j,i}) \wedge (v_{j,i} \in V_i)$
- Найденное решение отправляется на вычислитель (аналитическую или имитационную модель) -> $f(\boldsymbol{X_k})$
- Оптимальное решение $\boldsymbol{Y} = \{\boldsymbol{X_1^*}, \boldsymbol{X_2^*}, \dots \boldsymbol{X_z^*}\}$, $f(\boldsymbol{X_i^*}) \in \psi, \forall i$, где ψ – область удовлетворительных значений целевой функции, обычно $f(\boldsymbol{X_i^*}) \leq B$, а $\psi = (-\infty; B]$



Параметрическая задача. Один вычислитель.

Время поиска N решений

- Последовательное выполнение: $t_{\text{об}} = N\left(t_{\text{поиск } x} + t_{\text{вычисление } f(x)}\right)$
- Параллельное выполнение $t_{\text{об}} = Nmax\left(t_{\text{поиск } x}; t_{\text{вычисление } f(x)}\right)$
  - $t_{\text{поиск } x} = t_{\text{вычисление } f(x)}$ - идеальный случай
  - $t_{\text{поиск } x} > t_{\text{вычисление } f(x)}$ - ускоряем поиск решений (метод оптимизации)
  - $t_{\text{поиск } x} < t_{\text{вычисление } f(x)}$ - ускоряем работу вычислителя
    - Применение промежуточного хранилища: хэш-таблица, база данных и т.д.
    - $t_{\text{вычисление } f(x)} = pt_{\text{поиск в хэш}-\text{таблице}} + (1-p)t_{\text{добавление в хэш}-\text{таблицу}}t_{\text{работы модели}}$
    - $t_{\text{поиск в хэш}-\text{таблице}} \ll t_{\text{работы модели}}$

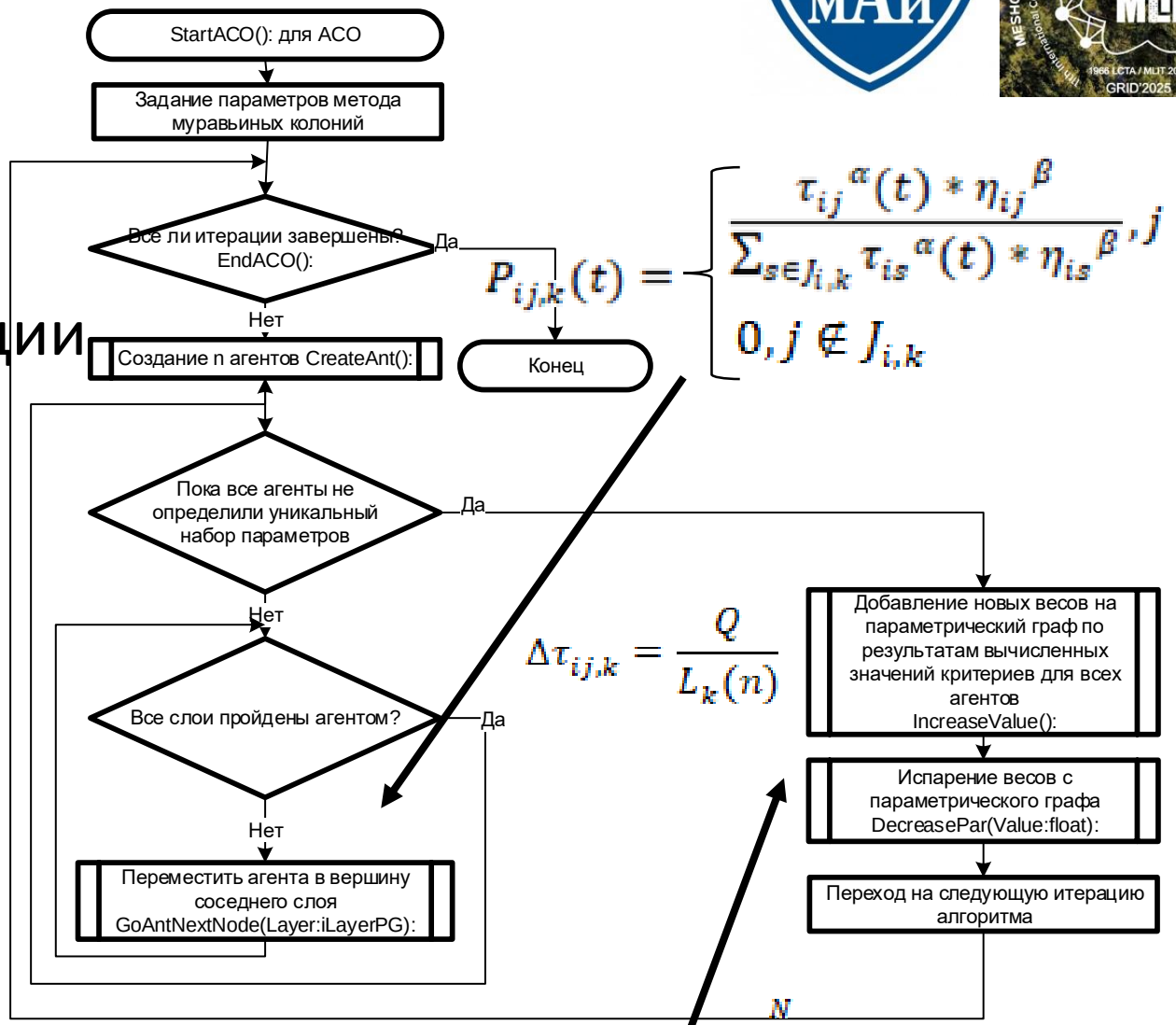


Метод муравьиных колоний. Классический.

- Задача TSP, QAP
- Стагнация алгоритма
- Наличие эвристической информации

Известные модификации:

- Ant Algorithms (1991)
- Elitist Ant System (1991)
- ANTQ (1995)
- Ant System (1997)
- AS на основе ранга (1997)
- MAX-MIN Ant System (2000)
- Best-Worst Ant System (2002)



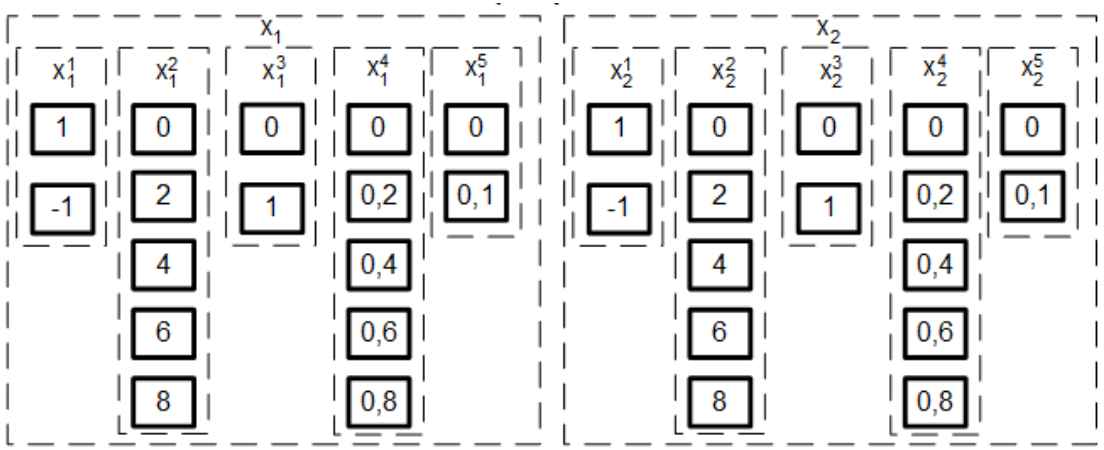
Метод муравьиных колоний. Параметрическая задача.

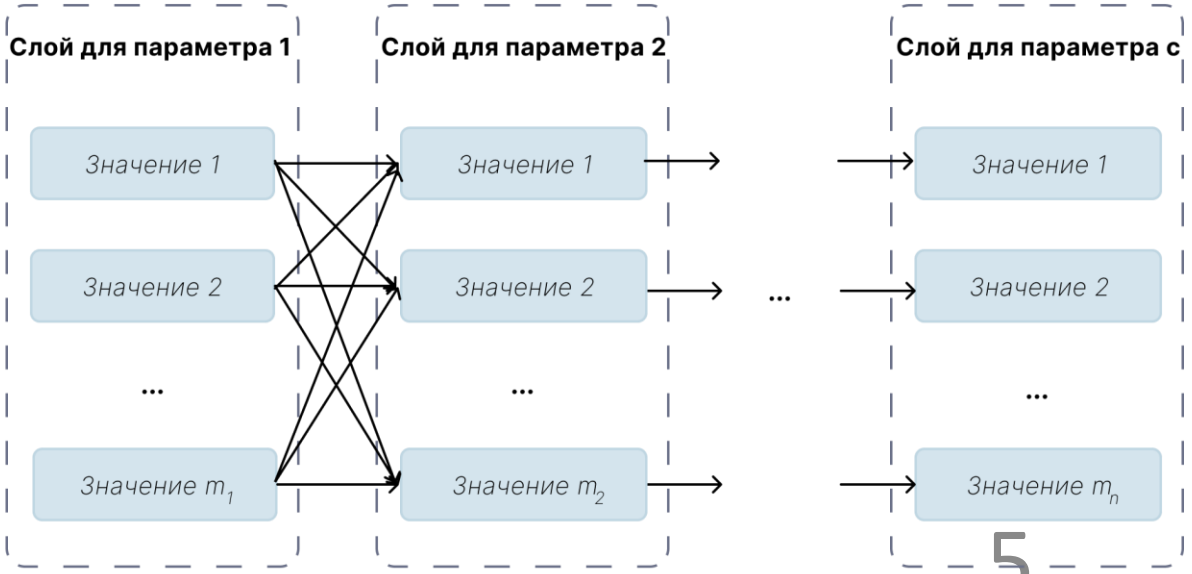
Требуемые модификации:

- Работа с отрицательными значениями целевой функции
- Новая вероятностная формула

$$z_{i,j}(t) = (\lambda_1 \tau^{\alpha}{}_{norm,i,j}(t) + \lambda_2 \left({}^{1}/_{\theta_{i,j}(t)}\right)^{\beta} + \lambda_3 \left({}^{\theta_{i,j}(t)}/_{\theta_{max,i}}\right)^{\gamma})H_{i,j}{}^{\delta}$$

- Параметрический граф
  - Оптимальная структура





Параллельный метод муравьиных колоний.

- P-ACO, Parallel ACO (2002)
- GMMAS (2009)
- распараллеливание уже существующих методов, разработанных для ACO (2011)
- ACS-GPU-SPM (2012)
- I-Roulette, DSRoulette (2013-2017)
- ACO с Xeon Phi (2014-2016)
- Параллельный ACS (2016)
- PartialACO (2017)
- ACS-GPU-Alt (2020)

- P-ACO – (Russell S.J. 2021, Norvig P. 2021, Abdelbar A. M. 2019)
- OpenMP – (Abouelfarag A.A. 2015, Mansour I.B., 2020)
- CUDA – (Tsutsui S. 2012, Cecilia J.M. 2013, Skinderowicz R. 2020)
- SSE и AVX – (Peake J. 2021, Tirado F. 2017, D´ıaz D.P. 2020)

Матричный метод муравьиных колоний.

- Дополнение параметрического графа незначимыми вершинами до $m = \max(m_i)$
- Входные матрицы $\boldsymbol{V}, \boldsymbol{T}, \boldsymbol{\theta} = (n \times m)$;

Подготовка данных

- $\boldsymbol{Z} = \lambda_1 \boldsymbol{T_{norm}} + \lambda_2 ({}^{1}/_{\boldsymbol{\theta}}) + \lambda_3 ({}^{\boldsymbol{\theta}}/_{\boldsymbol{\theta}'}) \; \boldsymbol{Z} = (n \times m)$
- $\boldsymbol{P} = {}^{\boldsymbol{Z}}/_{\boldsymbol{Z}'}, \boldsymbol{P} = (n \times m)$
- $\boldsymbol{F} = (n \times m), \boldsymbol{F}_{i,j} = \sum_{k=1}^{j} \boldsymbol{P}_{i,j}$

Модификация ACO

- Генерируется матрица реализации случайных величин, равномерно распределенных в интервале (0; 1) $\boldsymbol{R} = (K \times n)$
- Определяется позиция s $\boldsymbol{F}_{i,s-1} \le r_{k,i} \le \boldsymbol{F}_{i,s}$

Занесение весов-феромона

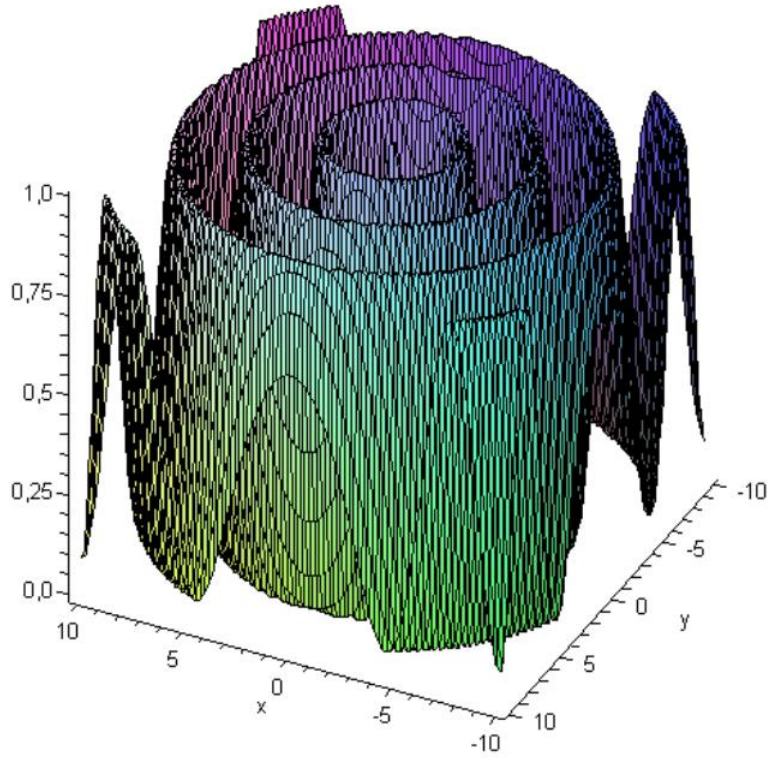
- $\boldsymbol{T}(\boldsymbol{t}+\boldsymbol{1}) = \rho \boldsymbol{T}(\boldsymbol{t})$
- $\boldsymbol{T}[\boldsymbol{X}[k,j],j](t+1) = \boldsymbol{T}[\boldsymbol{X}[k,j],j](t+1) + Q/\boldsymbol{Y}[k]$

Исследование на тестовых функциях.

- Функция Шаффера:

$$f(x_1, x_2) = \frac{1}{2} + \frac{sin^2\left(\sqrt{{x_1}^2 + {x_2}^2}\right) - 0.5}{1 + 0.001({x_1}^2 + {x_2}^2)}$$

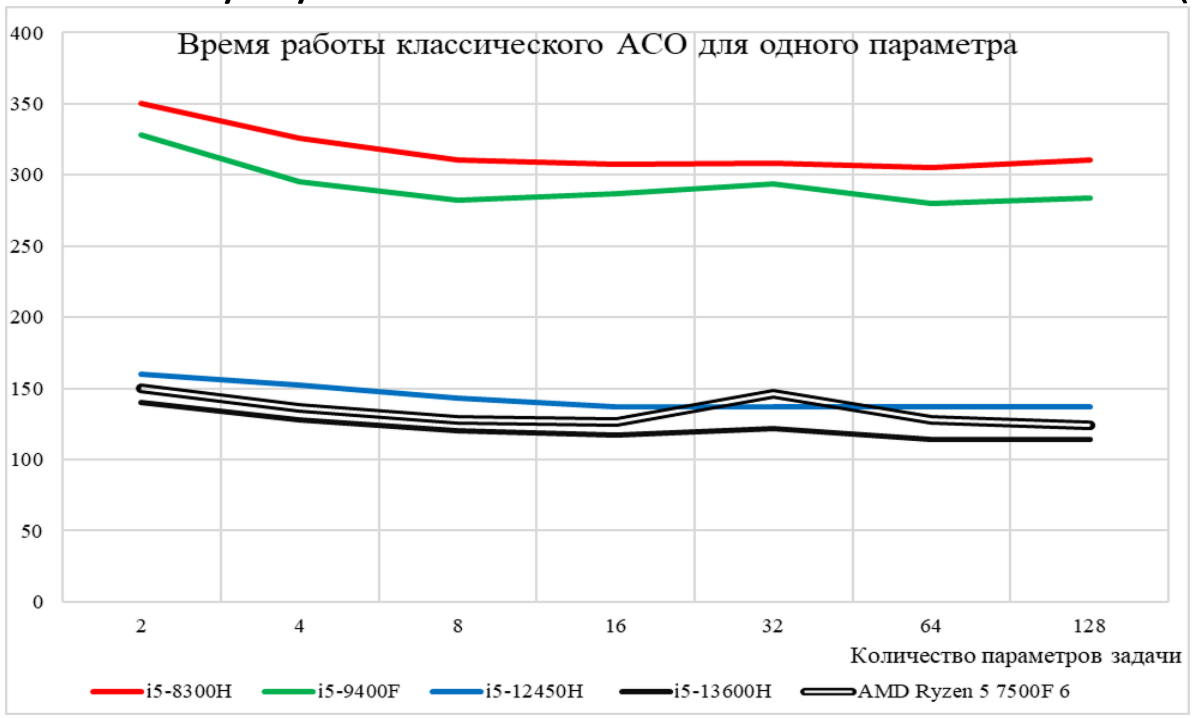
- на интервале [-10… 10] с точностью до 10^{-9}
- $m = 5$
  - 2 параметра ($n = 42$),
  - 4 параметра ($n = 84$),
  - 8 параметров ($n = 168$) ,
  - 16 параметров ($n = 336$),
  - 32 параметра ($n = 672$),
  - 64 параметра ($n = 1344$),
  - 128 параметров ($n = 2688$)
- $\lambda_1, \lambda_2, \alpha, \beta = 1,\ Q = 1, \rho = 0.999, K = 500$

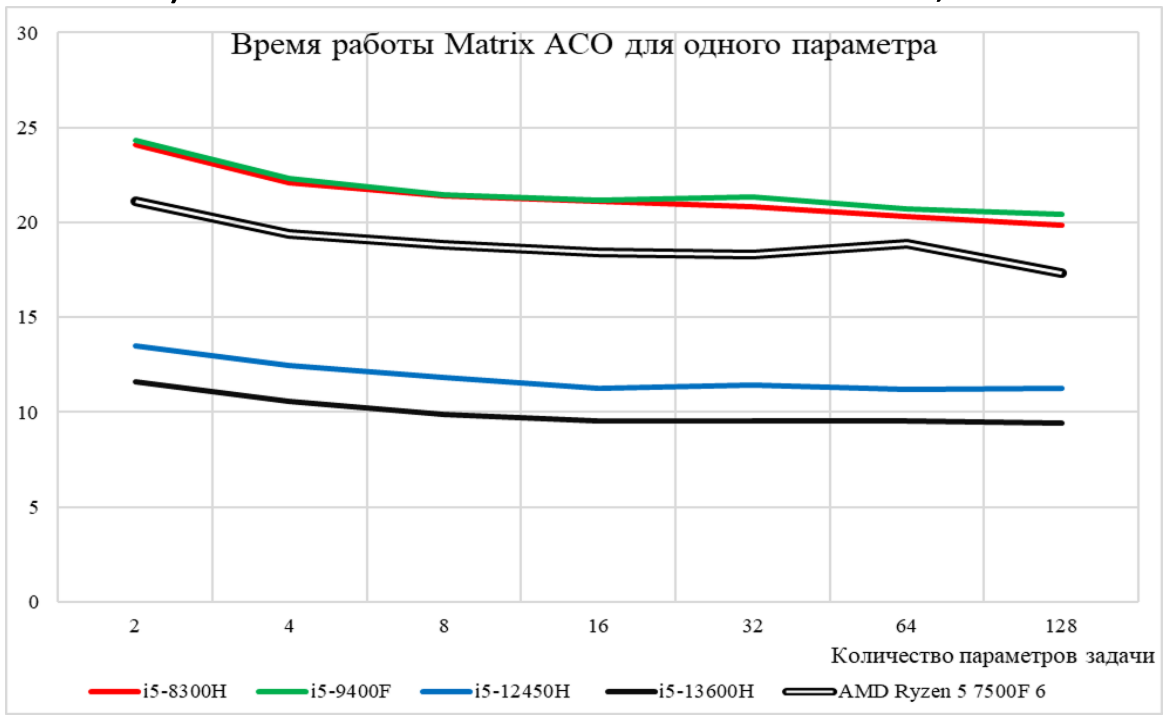


- Тестировались и другие тестовые функции: Растригина, Розенброка и т.д.

Аппаратное обеспечение.

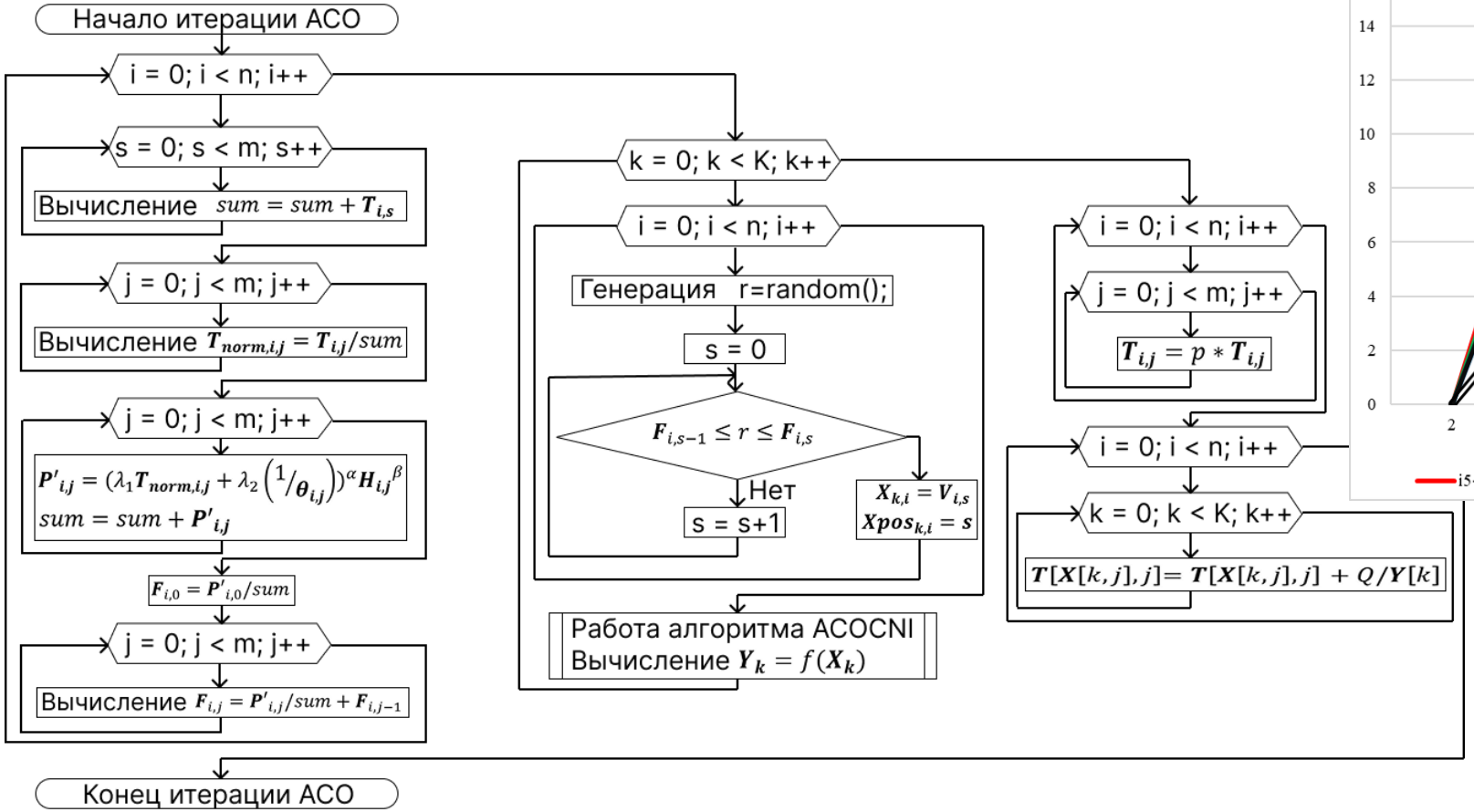
1. Сервер DELL POWEREDGE C4140 Intel Xeon Gold 6130 2.1 GHz 16 Core (2Tb ОЗУ) NVIDIA Tesla V100 16GB SXM2 (Вычислитель РЭУ)
2. Платформа для разработки приложений искусственного интеллекта NVIDIA Jetson Orin
3. ПК AMD Ryzen 5 7500F 6-Core Processor 3.70GHz (8Gb ОЗУ) + NVIDIA GeForce RTX 4060 Ti
4. ПК 13th Gen Intel Core i5-13600K 3.50 GHz (64Gb ОЗУ) + NVIDIA GeForce RTX 3060,
5. Ноутбук 12th Gen Intel Core i5-12450H 2.00 GHz (16Gb ОЗУ) + NVIDIA GeForce RTX 3060 Laptop GPU,
6. ПК 9th Gen Intel Core i5-9400F 2.90 GHz (16Gb ОЗУ) + NVIDIA GeForce GTX 1050 Ti,
7. Ноутбук 8th Gen Intel Core i5-8300H 2.30 GHz (8Gb ОЗУ) + NVIDIA GeForce GTX 1050 Ti;

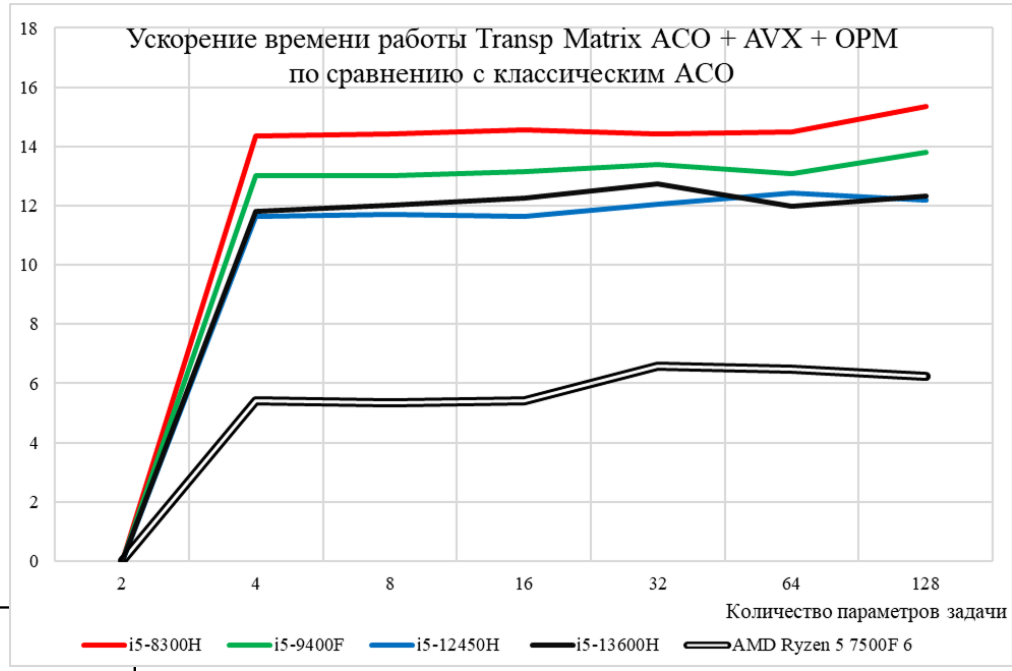




Транспонированный матричный метод муравьиных колоний. AVX

- Параллельное считывание инструкций по внутренним циклам матричного метода
- Размерность параметрического графа $m = 5$





Матричный метод муравьиных колоний. AVX

- Исследовались графы различной структуры, по 100, 1000, 10000 значений на один слой.
- 5 значений на слой – 21 слой и 33 фиктивные вершины
- 100 значений на слой – 6 слоев и 98 фиктивных вершин
- 1000 значений на слой – 5 слоев и 1998 фиктивных вершин
- 10000 значений на слой – 4 слоев и 19898 фиктивных вершин

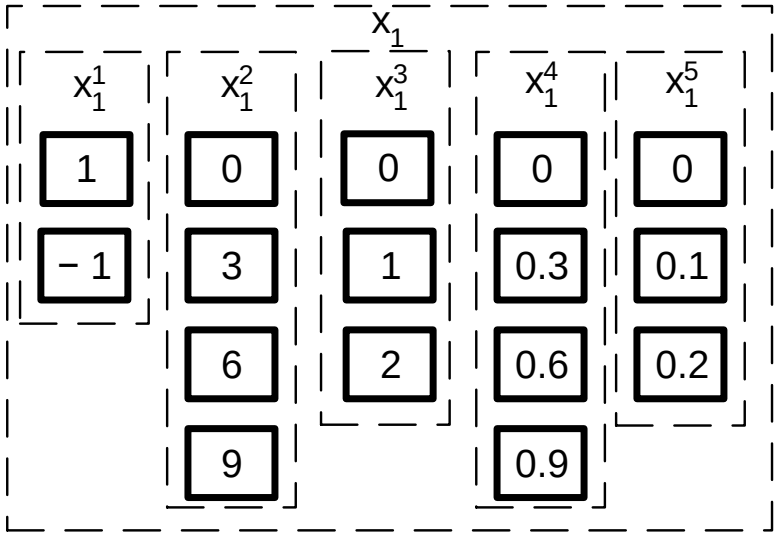
12th Gen Intel Core i5-12450H 2.00 GHz (16Gb ОЗУ)

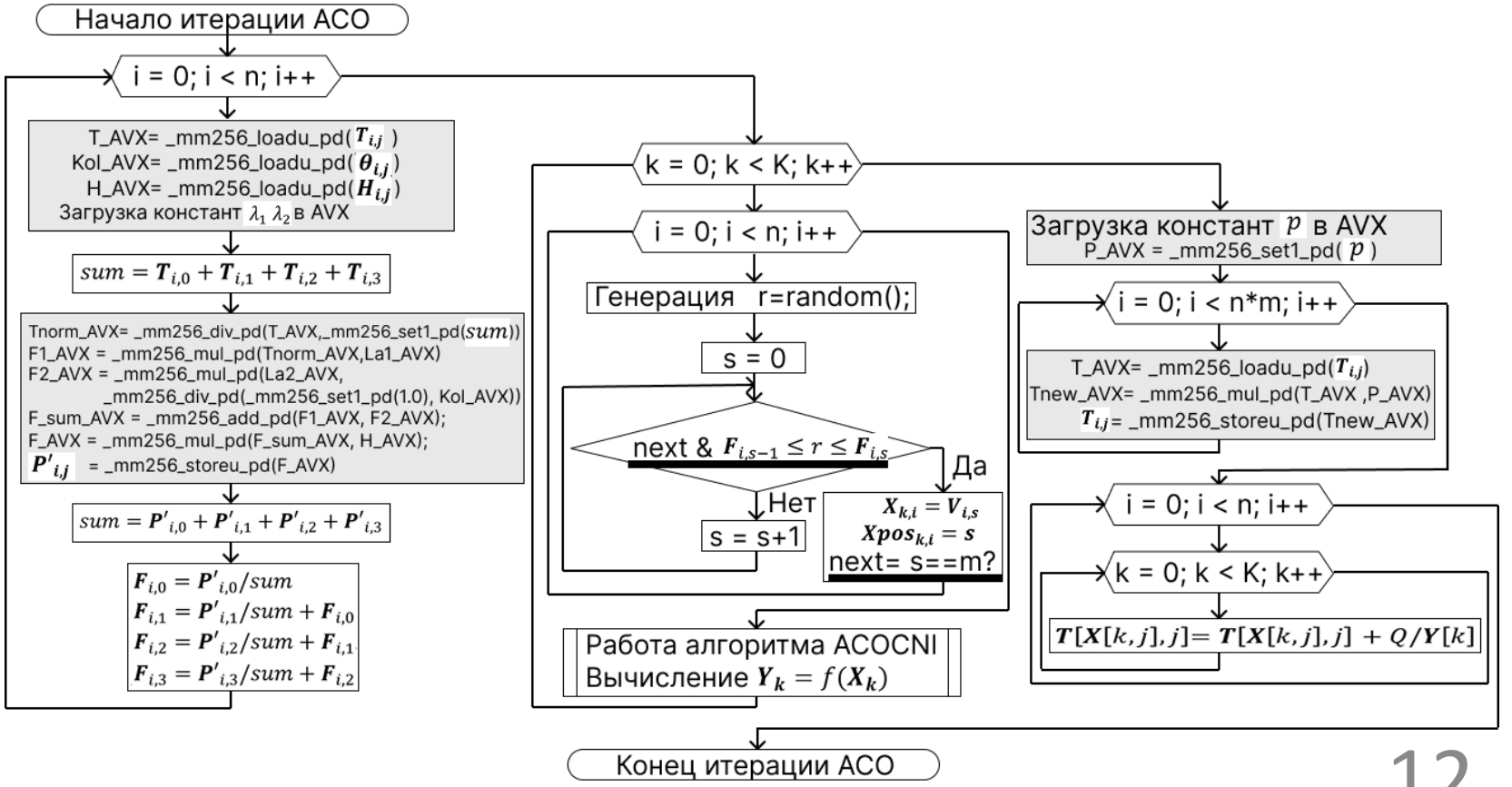
Время на один параметр (меньше-лучше)

m	Количество параметров		2	4	8	16	32	64	128
	Количество слоев графа		42	84	168	336	672	1344	2688
5	Classic ACO	МО ДИ	160,3 ±0,708	152,3 ±0,205	143,5 ±0,584	137,1 ±0,143	136,8 ±0,763	137,2 ±0,577	136,7 ±0,505
	Matrix ACO + OMP	МО ДИ	12,8 ±0,107	12,0 ±0,032	11,5 ±0,014	11,4 ±0,026	11,6 ±0,025	11,3 ±0,018	11,2 ±0,012
100	Classic ACO	МО ДИ	146,5 ±1,896	139,3 ±1,235	130,4 ±3,267	125,6 ±0,310	125,9 ±0,738	124,3 ±0,088	124,6 ±0,135
	Matrix ACO	МО ДИ	9,0 ±0,269	7,5 ±0,138	6,1 ±0,066	5,7 ±0,039	5,9 ±0,039	5,9 ±0,033	5,3 ±0,025
	Matrix ACO + OMP	МО ДИ	7,0 ±0,037	6,3 ±0,085	5,5 ±0,024	5,3 ±0,022	6,4 ±0,101	5,8 ±0,030	5,3 ±0,015
	Matrix ACO + AVX	МО ДИ	9,2 ±0,077	7,2 ±0,039	6,2 ±0,050	5,8 ±0,025	6,6 ±0,022	5,9 ±0,030	5,4 ±0,018
	Matrix ACO + AVX + OMP	**МО ДИ**	**6,2 ±0,033**	**5,6 ±0,020**	**5,2 ±0,010**	**5,3 ±0,012**	**6,0 ±0,025**	**5,2 ±0,024**	**5,8 ±0,020**
	Transp Matrix ACO	МО ДИ	8,6 ±0,083	7,2 ±0,070	6,2 ±0,075	5,7 ±0,015	6,7 ±0,025	5,7 ±0,011	6,2 ±0,011
	Transp Matrix ACO + AVX	МО ДИ	9,4 ±0,091	7,5 ±0,059	6,4 ±0,037	6,1 ±0,017	6,8 ±0,019	5,8 ±0,019	6,2 ±0,012
	Transp Matrix ACO + AVX + OPM	МО ДИ	9,4 ±0,096	7,5 ±0,044	6,5 ±0,048	6,1 ±0,017	6,8 ±0,020	5,8 ±0,012	6,2 ±0,010

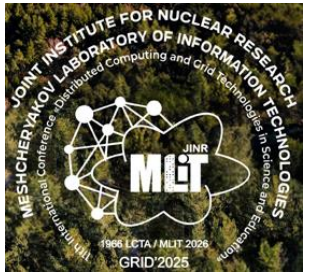
Матричный метод муравьиных колоний. Специальный граф с m=4 для AVX.

- Количество вершин определяется максимальным дискретным числом и разложением на простые сомножители.
- Для 10, 2 слоя: 2 и 5
- Для 12, 2 слоя; 3 и 4





Матричный метод муравьиных колоний. Специальный граф с m=4 для AVX.



9th Gen Intel Core i5-9400F 2.90 GHz (16Gb ОЗУ)

Количество параметров	2	4	8	16	32	64	128
Количество слоев	42	84	168	336	672	1344	2688
Матричный ACO	12,1	12,2	15,8	12,3	12,8	11,9	12,1
Матричный ACO + OMP	12,9	12,5	16,2	12,5	13,0	12,0	12,3
Транспонированный матричный ACO	11,9	12,0	15,8	12,4	12,9	12,0	12,4
Транспонированный матричный ACO + AVX		11,8	15,6	12,2	12,7	12,0	12,3
Транспонированный матричный ACO + AVX + OMP		11,8	15,6	12,3	12,7	12,0	12,3
Матричный ACO + AVX + OMP + специальный граф	11,8	12,5	16,4	12,9	13,4	12,6	13,1

AMD Ryzen 5 7500F 6-Core Processor 3.70GHz (8Gb ОЗУ)

Ускорение относительно классического метода (больше-лучше)

Количество параметров	2	4	8	16	32	64	128
Количество слоев	42	84	168	336	672	1344	2688
Матричный ACO	7,1	7,0	6,8	6,9	8,0	6,8	7,2
Матричный ACO + OMP	7,5	7,3	7,0	6,9	8,1	7,2	7,1
Транспонированный матричный ACO	7,1	7,0	6,8	6,9	8,0	7,1	7,1
Транспонированный матричный ACO + AVX		5,4	5,3	5,4	6,6	6,5	6,4
Транспонированный матричный ACO + AVX + OMP		5,4	5,3	5,4	6,6	6,5	6,2
Матричный ACO + AVX + OMP + специальный граф	5,6	5,5	5,4	5,5	7,0	6,4	6,0

Матричный метод муравьиных колоний. CUDA

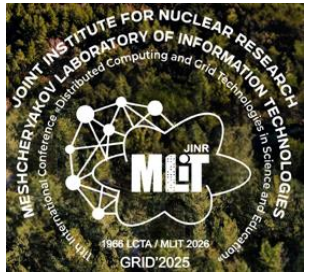
- Исполнение этапов 1,2 и 3 на GPU, объединение этапа 1 и 3, объединение всех этапов на GPU.
- На GPU определялась возможность использования константной памяти, локальной памяти или хранения всех данных в глобальной памяти GPU
- Различное количество потоков и блоков для технологии CUDA.

NVIDIA Tesla V100 16GB SXM2

Ускорение относительно классического метода (больше-лучше)

Количество параметров	2	4	8	16	32	64	128
Количество слоев	42	84	168	336	672	1344	2688
CUDA отдельно 1,2,3 этапы	34,35	45,57	58,02	59,20			
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы	33,63	46,38	59,10	70,37	76,76	75,67	73,39
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + параметры	32,67	42,86	55,19	61,70	60,85	61,68	69,91
CUDA совместно 1+3 этапы и отдельно 2-ой	34,73	46,36	58,01	59,43			
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам)	34,45	47,06	59,36	70,76	76,73	75,98	73,74
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + параметры	33,09	43,77	55,28	62,10	61,00	61,11	69,97
CUDA совместно 1+2+3 этапы	2,42	3,43	7,60	7,68	13,47		
CUDA совместно 1+2+3 этапы (параллелизм только по муравьям-агентам)	1,42	1,17	1,05	1,03	0,98	0,90	0,86
CUDA совместно 1+2+3 этапы (параллелизм только по муравьям-агентам) + локальные переменные		47,18	42,47	36,24	34,10	31,89	
CUDA отдельно 1,2,3 этапы + константная память	35,63	48,42	59,74	59,66	40,18		
CUDA отдельно 1,2(параллелизм только по муравьям-агентам),3 этапы + константная память	35,85	50,42	65,58	80,58	83,28	84,72	
CUDA отдельно 1,2(параллелизм только по муравьям-агентам),3 этапы + константная память + параметры	34,77	46,80	59,79	69,11	65,02	67,16	
CUDA совместно 1+3 этапы и отдельно 2-ой + константная память	36,50	49,27	60,62	59,91	40,28		
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + константная память	36,66	51,35	66,42	81,32	83,79	84,52	
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + константная память + параметры	35,64	47,72	60,91	69,76	65,38	66,98	
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + только глобальная память	31,36	39,44	47,53	51,33	51,80	53,34	54,18
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + только глобальная память	32,00	40,11	48,09	51,72	52,00	53,41	54,22

Матричный метод муравьиных колоний. CUDA



NVIDIA GeForce RTX 4060 Ti

Количество параметров	2	4	8	16	32	64	128
Количество слоев	42	84	168	336	672	1344	2688
CUDA отдельно 1,2,3 этапы	17,23	21,53	14,49	8,32			
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы	12,23	17,68	23,66	28,60	37,88	33,78	33,40
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + параметры	12,23	17,84	23,79	29,22	38,05	36,88	37,46
CUDA совместно 1+3 этапы и отдельно 2-ой	17,17	22,13	14,81	8,57			
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам)	12,50	18,06	24,00	29,58	37,41	34,08	33,68
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + параметры	12,41	17,99	24,20	29,66	37,52	36,22	37,50
CUDA совместно 1+2+3 этапы	12,27	13,18	8,45	4,76			
CUDA совместно 1+2+3 этапы (параллелизм только по муравьям-агентам)	0,43	0,40	0,37	0,37	0,43	0,37	0,36
CUDA совместно 1+2+3 этапы (параллелизм только по муравьям-агентам) + локальные переменные	8,21	7,75	7,38	7,28	8,36	6,77	5,10
CUDA отдельно 1,2,3 этапы + константная память	17,08	22,12	14,75	8,56			
CUDA отдельно 1,2(параллелизм только по муравьям-агентам),3 этапы + константная память	12,37	18,07	23,73	29,55	38,97	33,78	
CUDA отдельно 1,2(параллелизм только по муравьям-агентам),3 этапы + константная память + параметры	12,34	17,98	24,03	29,69	39,09	36,42	
CUDA совместно 1+3 этапы и отдельно 2-ой + константная память	17,39	22,27	14,78	8,64			
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + константная память	12,63	18,08	23,88	29,63	39,07	33,80	
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + константная память + параметры	12,38	18,03	24,07	29,71	39,15	36,75	
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + только глобальная память	12,37	17,77	23,66	29,10	38,55	36,57	37,12
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + только глобальная память	12,17	17,92	23,94	29,24	37,13	36,32	37,12

NVIDIA Jetson Orin

Количество параметров	2	4	8	16	32	64	128
Количество слоев	42	84	168	336	672	1344	2688
CUDA отдельно 1,2,3 этапы	11,65	10,40	14,86	11,16			
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы	15,04	17,55	18,45	19,34	18,24	17,99	18,11
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + параметры	16,61	23,70	33,63	38,18	37,34	38,83	41,39
CUDA совместно 1+3 этапы и отдельно 2-ой	11,68	10,42	14,89	11,15			
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам)	15,20	17,69	19,29	19,47	18,45	17,89	18,10
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + параметры	16,77	23,85	33,90	38,40	37,09	39,14	41,37
CUDA совместно 1+2+3 этапы							
CUDA совместно 1+2+3 этапы (параллелизм только по муравьям-агентам)							
CUDA совместно 1+2+3 этапы (параллелизм только по муравьям-агентам) + локальные переменные							
CUDA отдельно 1,2,3 этапы + константная память	15,98	15,71	14,62	11,12			
CUDA отдельно 1,2(параллелизм только по муравьям-агентам),3 этапы + константная память	16,89	19,55	18,41	19,30	18,47	18,39	
CUDA отдельно 1,2(параллелизм только по муравьям-агентам),3 этапы + константная память + параметры	17,85	25,24	33,43	37,92	38,03	40,98	
CUDA совместно 1+3 этапы и отдельно 2-ой + константная память	16,08	15,77	14,85	11,14			
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + константная память	17,16	19,77	19,21	19,41	18,70	18,24	
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + константная память + параметры	18,08	25,52	33,65	38,10	38,22	41,13	
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + только глобальная память	16,72	23,34	29,96	32,41	31,97	33,43	35,33
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + только глобальная память	16,92	23,52	30,13	32,60	31,83	33,55	35,36

Матричный метод муравьиных колоний. Гетерогенный вычислитель

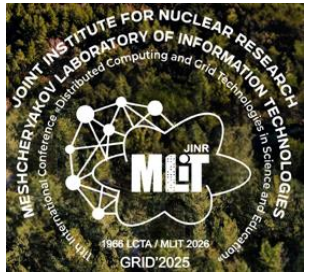
- Этапы 1 и 3 выполняются на CPU, этап 2 выполняется на GPU
- Постоянная передача информации по шине PCIe
- Применение графа с $m = 4$ и CUDA для второго этапа

Ускорение относительно классического метода (больше-лучше)

Количество параметров	2	4	8	16	32	64	128
Количество слоев	42	84	168	336	672	1344	2688
CUDA отдельно 1,2,3 этапы	17,23	21,53	14,49	8,32			
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы	12,23	17,68	23,66	28,60	37,88	33,78	33,40
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + параметры	12,23	17,84	23,79	29,22	38,05	36,88	37,46
CUDA отдельно 1,2,3 этапы + константная память	17,08	22,12	14,75	8,56			
CUDA отдельно 1,2(параллелизм только по муравьям-агентам),3 этапы + константная память	12,37	18,07	23,73	29,55	38,97	33,78	
CUDA отдельно 1,2(параллелизм только по муравьям-агентам),3 этапы + константная память + параметры	12,34	17,98	24,03	29,69	39,09	36,42	
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + только глобальная память	12,37	17,77	23,66	29,10	38,55	36,57	37,12
Гетерогенный алгоритм	24,82	25,74	26,93	27,27	31,16	28,55	28,02
Гетерогенный алгоритм + константная память	23,59	25,06	25,79	27,33	32,50	28,07	
Гетерогенный алгоритм + только глобальная память	24,25	24,64	26,43	26,67	31,10	27,71	27,87
Гетерогенный алгоритм + транспонированные матрицы + AVX		26,59	26,38	27,41	32,60	28,04	28,04
Инвертированный гетерогенный алгоритм	4,77	3,14	3,48	4,05	6,02	6,07	6,85
Матричный метод на специальном AVX графе	5,55	5,47	5,42	5,52	7,00	6,38	5,97
Гетерогенный алгоритм + специальный AVX граф	24,46	25,92	26,33	27,48	31,28	28,52	28,13

NVIDIA GeForce RTX 4060 Ti

Матричный метод муравьиных колоний. CUDA с большой размерностью.



NVIDIA Tesla V100 16GB SXM2

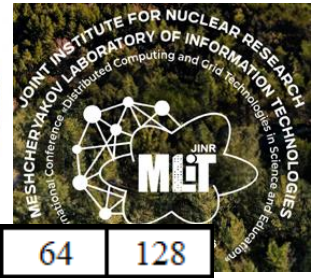
Время на один параметр (меньше-лучше)

Количество параметров	2	4	8	16	32	64	128	256
Количество слоев	42	84	168	336	672	1 344	2 688	5 376
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + только глобальная память	3,80	2,56	1,93	1,76	1,69	1,55	1,47	1,44
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + только глобальная память	3,72	2,52	1,91	1,75	1,69	1,55	1,47	1,44
Гетерогенный алгоритм + только глобальная память	4,10	3,50	3,36	3,69	3,94	3,98	3,92	3,91

Количество параметров	512	1024	2048	4096	8192	16384	32768	65536
Количество слоев	10 752	21504	43008	86016	172032	344064	688128	1376256
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + только глобальная память	1,42	1,42	1,42	1,44	1,46	1,61	1,47	1,63
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + только глобальная память	1,42	1,42	1,42	1,44	1,43	1,51	1,43	1,59
Гетерогенный алгоритм + только глобальная память	4,58	5,30	5,36	5,39	5,41			

	T_start	T_Delt	T1	T2	T3	Сумма	T2 в %
CUDA отдельно 1,2 (параллелизм только по муравьям-агентам),3 этапы + только глобальная память	0,0014	0,0036	0,0001	1,5731	0,0532	1,6315	96,42%
CUDA совместно 1+3 этапы и отдельно 2-ой (параллелизм только по муравьям-агентам) + только глобальная память	0,0040	0,0037	0,0297	1,5556	0,0000	1,5930	97,65%

Матричный метод муравьиных колоний.



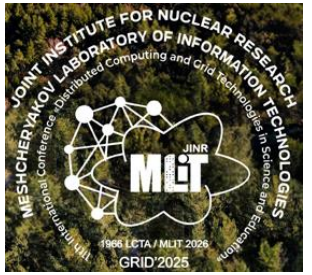
Вычислитель

1. Сервер DELL POWEREDGE C4140 Intel Xeon Gold 6130 2.1 GHz 16 Core (2Tb ОЗУ) + NVIDIA Tesla V100 16GB SXM2
2. Платформа для разработки приложений искусственного интеллекта NVIDIA Jetson Orin
3. ПК AMD Ryzen 5 7500F 6-Core Processor 3.70GHz (8Gb ОЗУ). + NVIDIA GeForce RTX 4060 Ti
4. ПК 13th Gen Intel Core i5-13600K 3.50 GHz (64Gb ОЗУ) + NVIDIA GeForce RTX 3060,
5. Ноутбук 12th Gen Intel Core i5-12450H 2.00 GHz (16Gb ОЗУ) + NVIDIA GeForce RTX 3060 Laptop GPU,
6. ПК 9th Gen Intel Core i5-9400F 2.90 GHz (16Gb ОЗУ) + NVIDIA GeForce GTX 1050 Ti,
7. Ноутбук 8th Gen Intel Core i5-8300H 2.30 GHz (8Gb ОЗУ) + NVIDIA GeForce GTX 1050 Ti;

Время на один параметр (меньше-лучше)

	Кол-во параметров	2	4	8	16	32	64	128
Ядра CUDA	Кол-во слоев	42	84	168	336	672	1 344	2 688
№ПК	Матричная модификация							
5120	1 Оптимальная модификация + константная память	3,25	1,97	1,38	1,11	1,05	0,98	
	1 Оптимальная модификация	3,46	2,15	1,55	1,28	1,14	1,09	1,08
1024	2 Оптимальная модификация	11,32	7,29	4,95	4,23	4,33	4,07	3,89
	2 Оптимальная модификация + константная память	10,51	6,82	4,99	4,26	4,21	3,87	
4352	3 AVX модификация на CPU для специального графа	6,12	5,26	4,87	4,61	4,68	4,48	4,42
	3 Оптимальная модификация	12,07	7,57	5,3	4,27	3,9	3,53	3,31
3584	4 Гетерогенная модификация с ускорением AVX для транспонированных матриц		6,67	6,1	6,1	6,01	5,8	5,74
	4 Классическая модификация	17,61	11,36	7,97	6,57	5,91	5,33	5,06
3840	5 Гетерогенная модификация + глобальная память	7,52	6,33	6,02	5,87	5,97	5,91	5,88
	5 Классическая модификация + глобальная память	16,57	10,58	7,55	6,02	5,45	5,07	4,88
768	6 AVX модификация на CPU для специального графа	13,94	13,03	12,62	12,46	14,07	14,21	14,51
	6 Классическая модификация	28,42	20,46	15,6	13,63	10,97	11,41	11,15
768	7 Гетерогенная модификация	14,71	13,4	12,94	12,8	12,8	12,69	12,8
	7 Оптимальная модификация	26,26	18,24	14,15	12,3	11,4	10,8	10,54

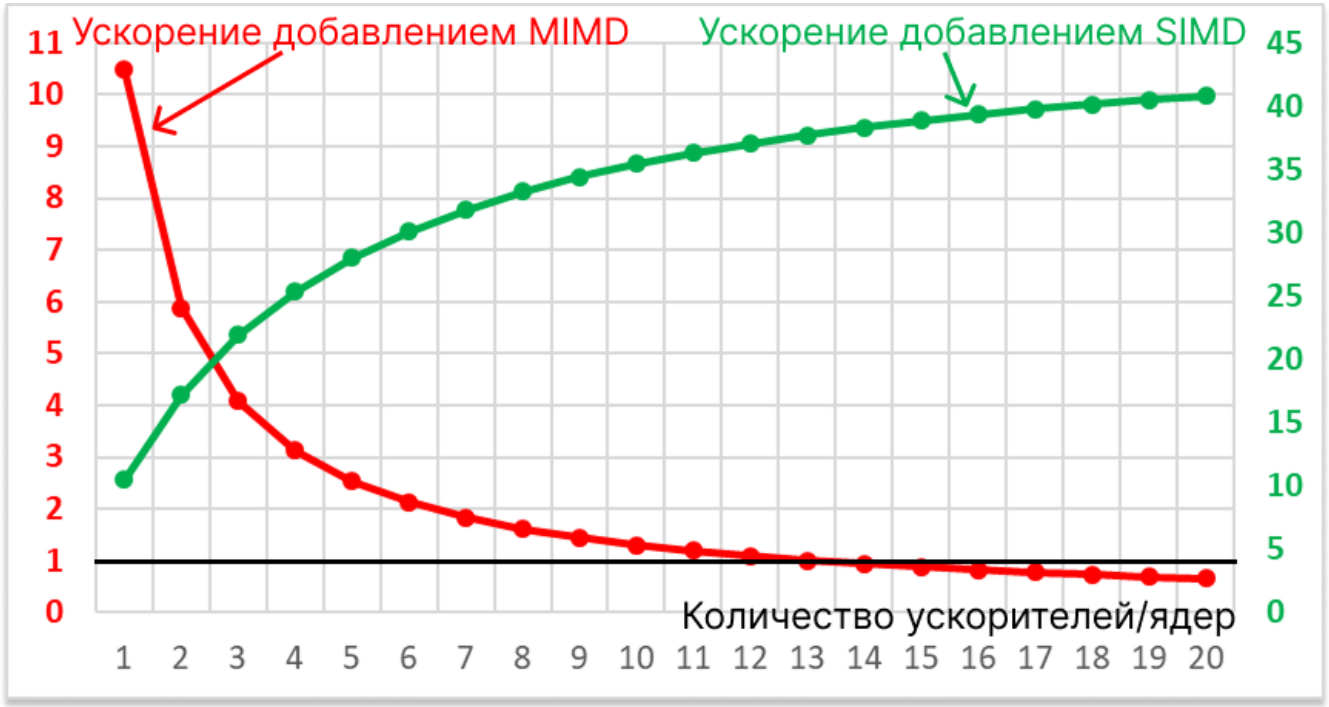
Подсчет коэффициентов ускорения гетерогенного вычислителя.



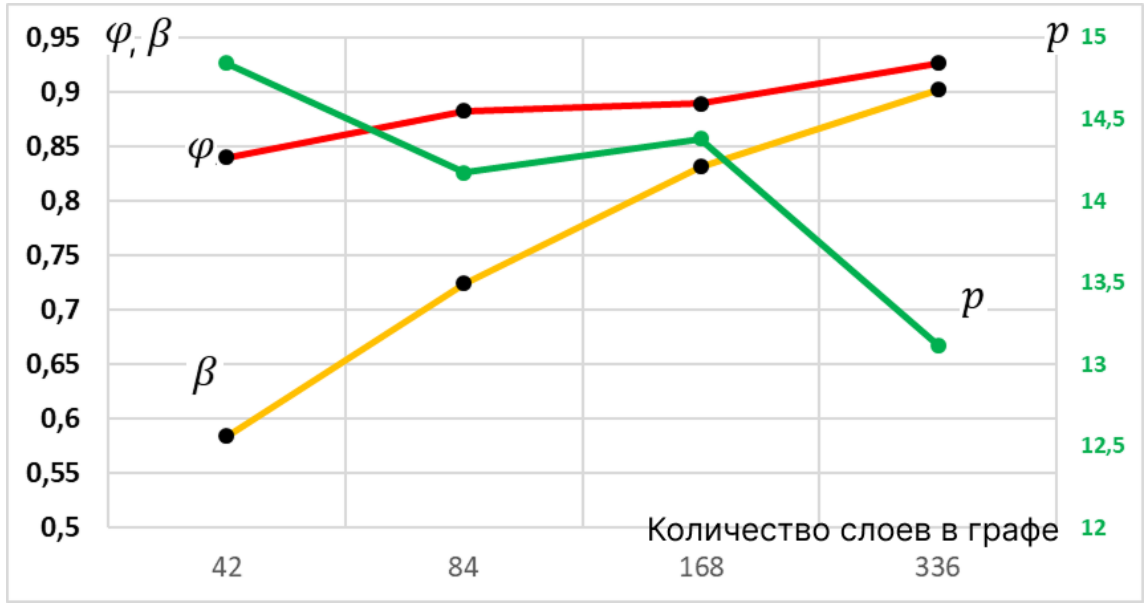
- $T_0 = T_{Start} + T_{Delt} + T_{Stage1} + T_{Stage2} + T_{Stage3}$
- $T_1 = T_0 - (T_{Start} + T_{Delt})$
- $T_{1,1} = T_S + T_M$
- $T_S = T_{S1} + T_{S2,NON\ CUDA} + T_{S3}; T_M = T_{SM} - T_{S2,NON\ CUDA};$
- Количество MIMD компонент – q; Количество SIMD компонент – r
- Доля MIMD компоненты – $\varphi = T_{M,Classic\ ACO}/T_1$
- Ускорение SIMD вычислителя – $\rho = T_{S,Classic\ ACO}/T_S$
- $K_q = \frac{T_q^1}{T_q^2} = \frac{T_1}{qT_1\left(\frac{(1-\varphi)}{\rho}+\frac{\varphi}{k}\right)} = \frac{k\rho}{q((1-\varphi)k+\varphi\rho)}; T_q^1 = T_1; T_q^2 = qT_{q,1}$
- $K_q = \frac{k_q\rho}{q((1-\varphi)k_q+\varphi\rho)}, K_r = \frac{k_r\rho}{k_r\rho\varphi+1-\varphi}, K_{q,r} = \frac{\rho k_r k_q}{q((1-\varphi)k_q+\varphi\rho k_r)}$

Структура гетерогенного вычислителя.

Пример для параметрической задачи с 16-ю параметрами



Зависимость коэффициентов



Подсчет коэффициентов ускорения гетерогенного вычислителя.

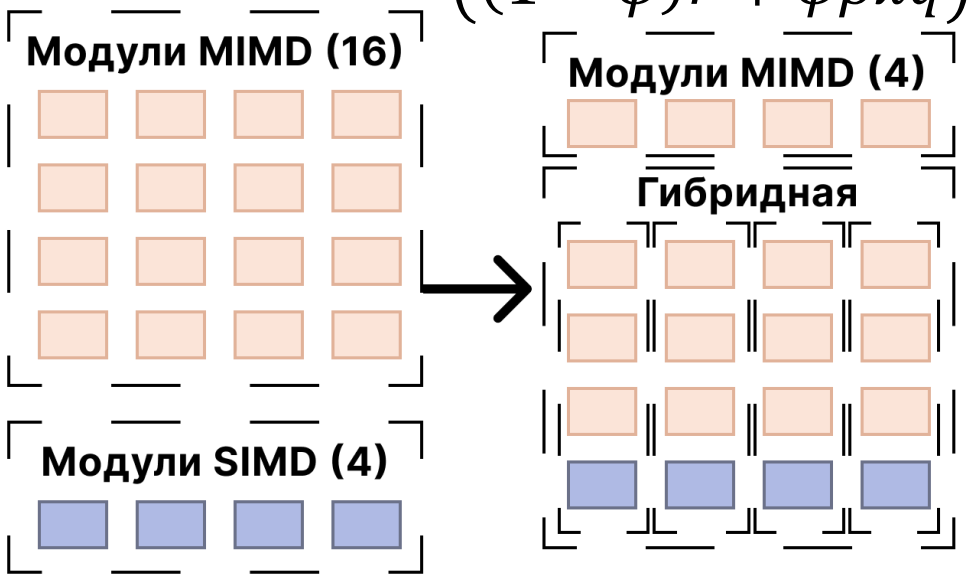
- Однородной MIMD структуры из $(q - q')$ универсальных ядер без SIMD ускорителей
- Гибридной структуры, содержащей q' MIMD ядер и r SIMD ускорителей, в которой с одним ускорителем взаимодействуют $d = \frac{q'}{r}$ ядер.

- $K_{q,d}^{re} = q - q' + r\frac{\rho d}{(1-\varphi)+\varphi\rho d} = q - q' + \frac{\rho q'}{(1-\varphi)+\frac{\varphi\rho q'}{r}} = q - q' + \frac{r\rho q'}{(1-\varphi)r+\varphi\rho q'};\ x = \frac{q'}{q}$

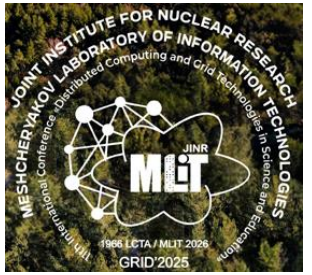
$$\frac{dK_{q,d}^{re}}{dx} = -q + \frac{rpq\big((1-\varphi)r + \varphi\rho xq\big) - r\rho qx(\varphi\rho q)}{\big((1-\varphi)r + \varphi\rho xq\big)^2} = -q + \frac{r^2pq(1-\varphi)}{\big((1-\varphi)r + \varphi\rho xq\big)^2} = 0 \Rightarrow$$

$$r^2pq(1-\varphi) = q\big((1-\varphi)r + \varphi\rho xq\big)^2 \Rightarrow$$

$$x = \frac{r(\sqrt{p(1-\varphi)}-(1-\varphi))}{\varphi\rho q};\ q' = \lfloor qx \rfloor$$



Учет времени передачи между оптимизатором и вычислителем.



- Оптимизатор связан с каждым вычислителем одной связью
  - $t_{\text{поиск } x} = 2t_{\text{передачи}} = t_{\text{вычисление } f(x)}$
- Оптимизатор связан с вычислителем в соответствии с топологией

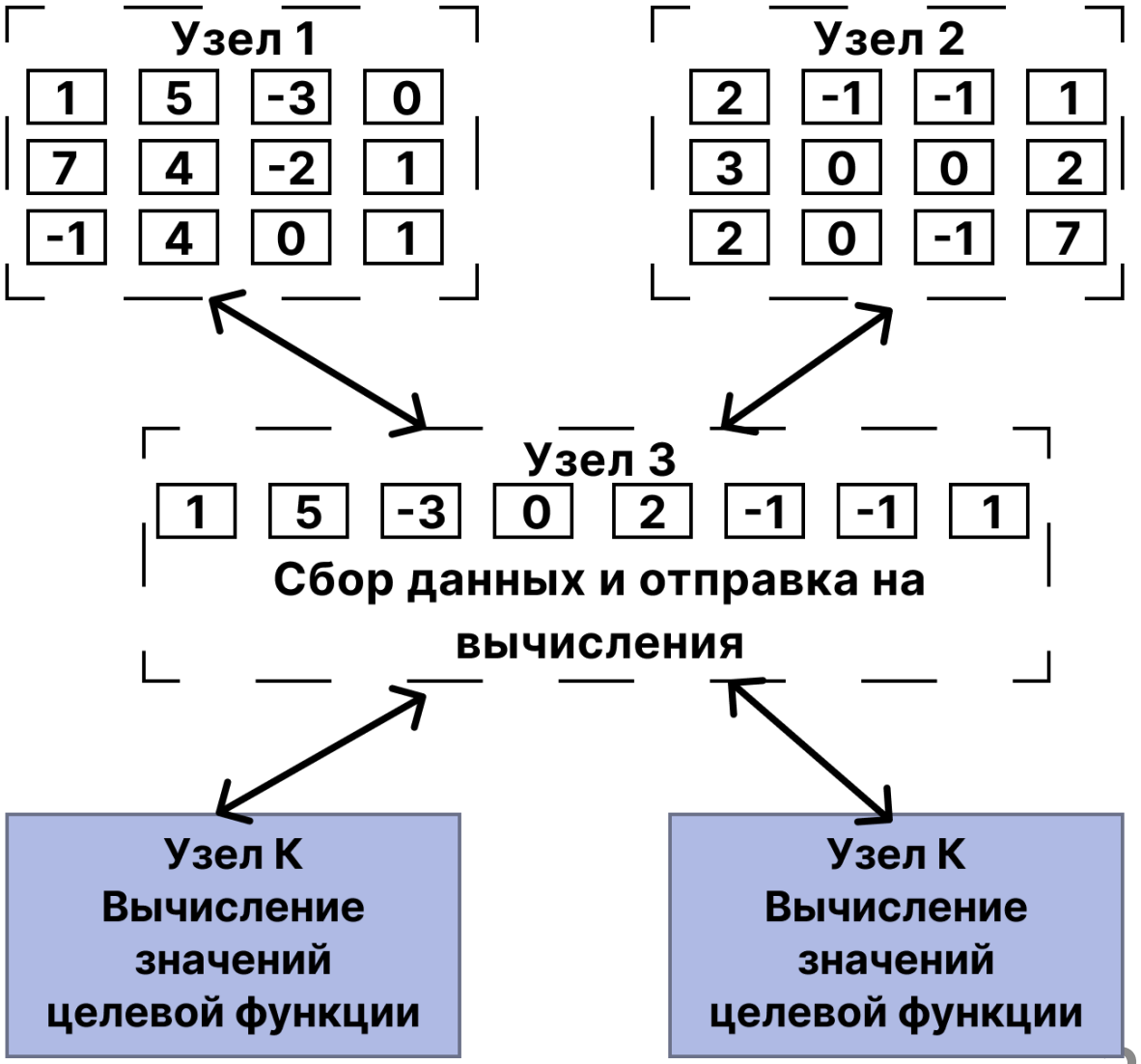
Параметр	1D-Tor	2D-Tor	3D-Tor	H	S	N
Максимальная длина	$\lceil \omega/2 \rceil$	$2\left\lceil \frac{\sqrt{\omega}}{2} \right\rceil$	$3\left\lceil \frac{\sqrt[3]{\omega}}{2} \right\rceil$	$\lceil \log(\omega) \rceil$	1	ω
Средняя длина	$\left\lceil \frac{\left(\frac{\omega}{2}-1\right)}{2} \right\rceil$	$\frac{\sqrt{\omega}}{2}$	$\frac{\sqrt{\omega}}{3}$	$\frac{\log(\omega)}{2}$	1	$\frac{\omega(\omega+1)}{2}$

Запуск метода муравьиных колоний на различных вычислителях.

- Зависимость от времени передачи
  - $K > \frac{t_{\text{передачи}}}{\varphi t\prime_2}$
- Предпочтительные компоновки

Время на один параметр (меньше-лучше)

	T_start	T_Delt	T1	T2	T3	T_OF	T_Hash
42	0,197	0,380	0,098	3,114	0,256	2,493	0,122
84	0,117	0,205	0,115	2,628	0,262	2,259	0,119
168	0,083	0,116	0,115	2,372	0,261	2,152	0,116
336	0,063	0,069	0,113	2,347	0,275	2,173	0,115
672	0,054	0,036	0,111	2,240	0,286	2,115	0,113
1344	0,049	0,031	0,103	2,259	0,297	2,144	0,113
2688	0,047	0,014	0,115	2,298	0,325	2,210	0,113



Параллельная матричная модификация метода муравьиных колоний для решения параметрических задач на гетерогенных вычислителях

Спасибо за внимание!

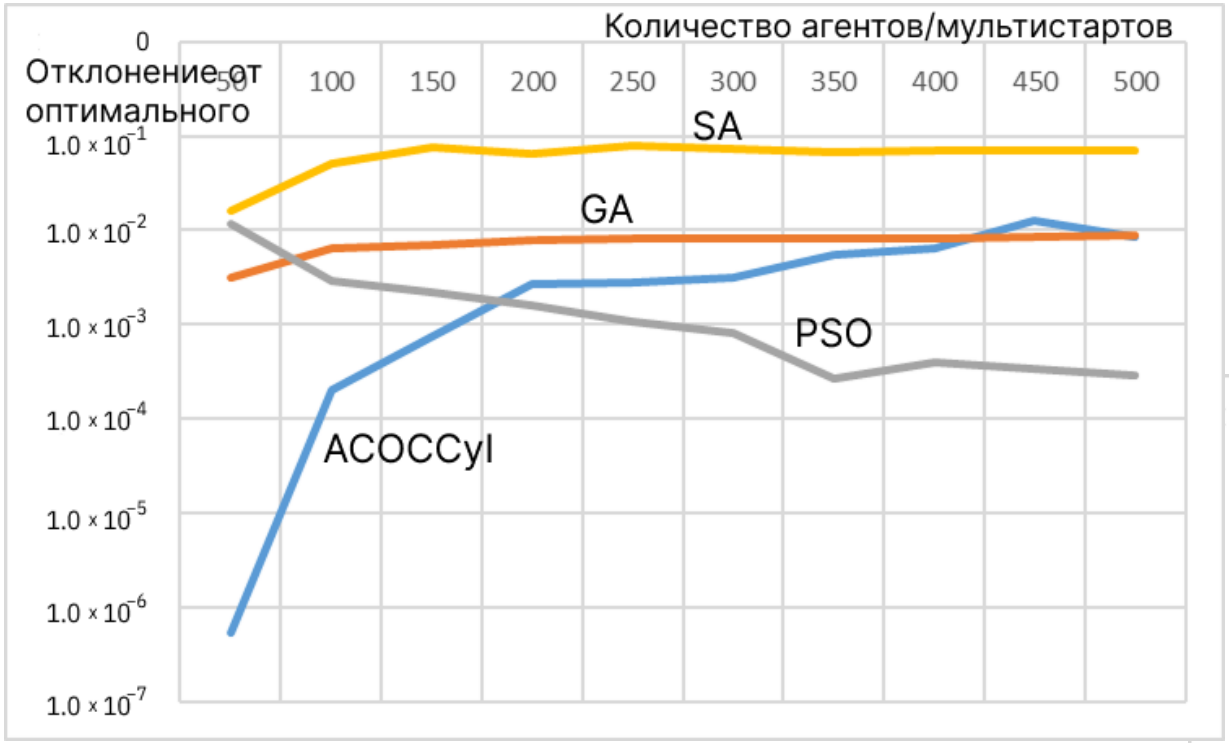
github.com/kalengul/ACO_SIMD

Титов Юрий Павлович, к.т.н.
kalengul@mail.ru
Судаков Владимир Анатольевич, д.т.н.
sudakov@ws-dss.com

Модификации метод муравьиных колоний.

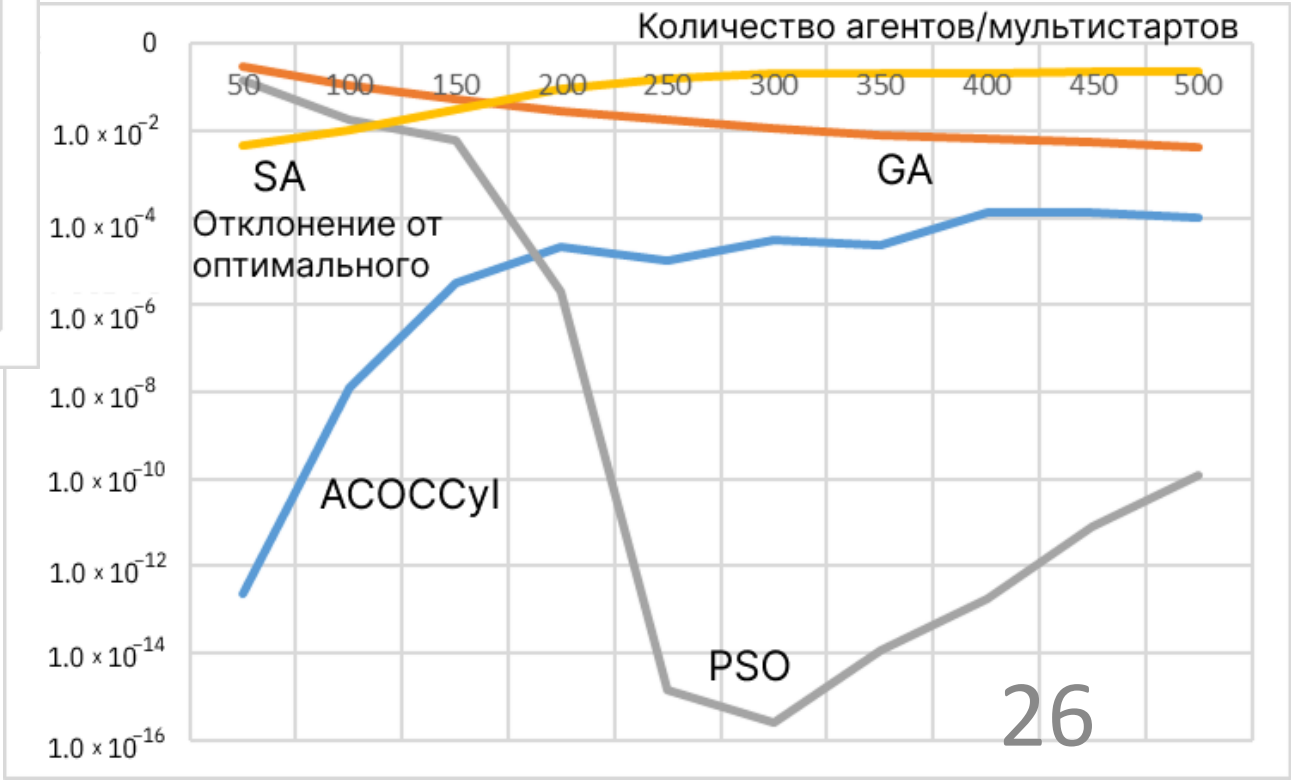
- ACOCN (ACO Cluster New) классический метод муравьиных колоний с применением хэш-таблицы и получением значений целевой функции без обращения к вычислителю. Если $\exists hash\_table(\boldsymbol{X}_k)$, то $\boldsymbol{Y_k} = hash\_table(\boldsymbol{X}_k)$
- **ACOCNI** (ACO Cluster New Ignor) если муравей-агент нашел уже рассмотренное решение, то данный муравей не заносит феромон на параметрический граф - игнорируется. Если $\exists hash\_table(\boldsymbol{X}_k)$, то $\boldsymbol{Y_k} = indefinitely$
- **ACOCCyN** (ACO Cluster Cycle N) если муравей-агент нашел уже рассмотренное решение, то производит дальнейший циклический поиск нового решения. Цикл ограничен N итераций, если новое решение не найдено – то муравей игнорируется.
- ACOCCyI (ACO Cluster Cycle Infinity) если муравей-агент нашел уже рассмотренное решение, то производит дальнейший циклический поиск пока не будет найдено новое решение.
- ACOCT (ACO Cluster Tree) Если муравей-агент нашел уже рассмотренное решение, то новое решение ищется алгоритмом неявного перебора, рассматривая параметрический граф как дерево.
- ACOCTSort (ACO Cluster Tree Sort) Если муравей-агент нашел уже рассмотренное решение, то новое решение ищется алгоритмом неявного перебора, рассматривая параметрический граф как дерево с сортировкой слоев.

Эффективность модификаций по сравнению с другими методами

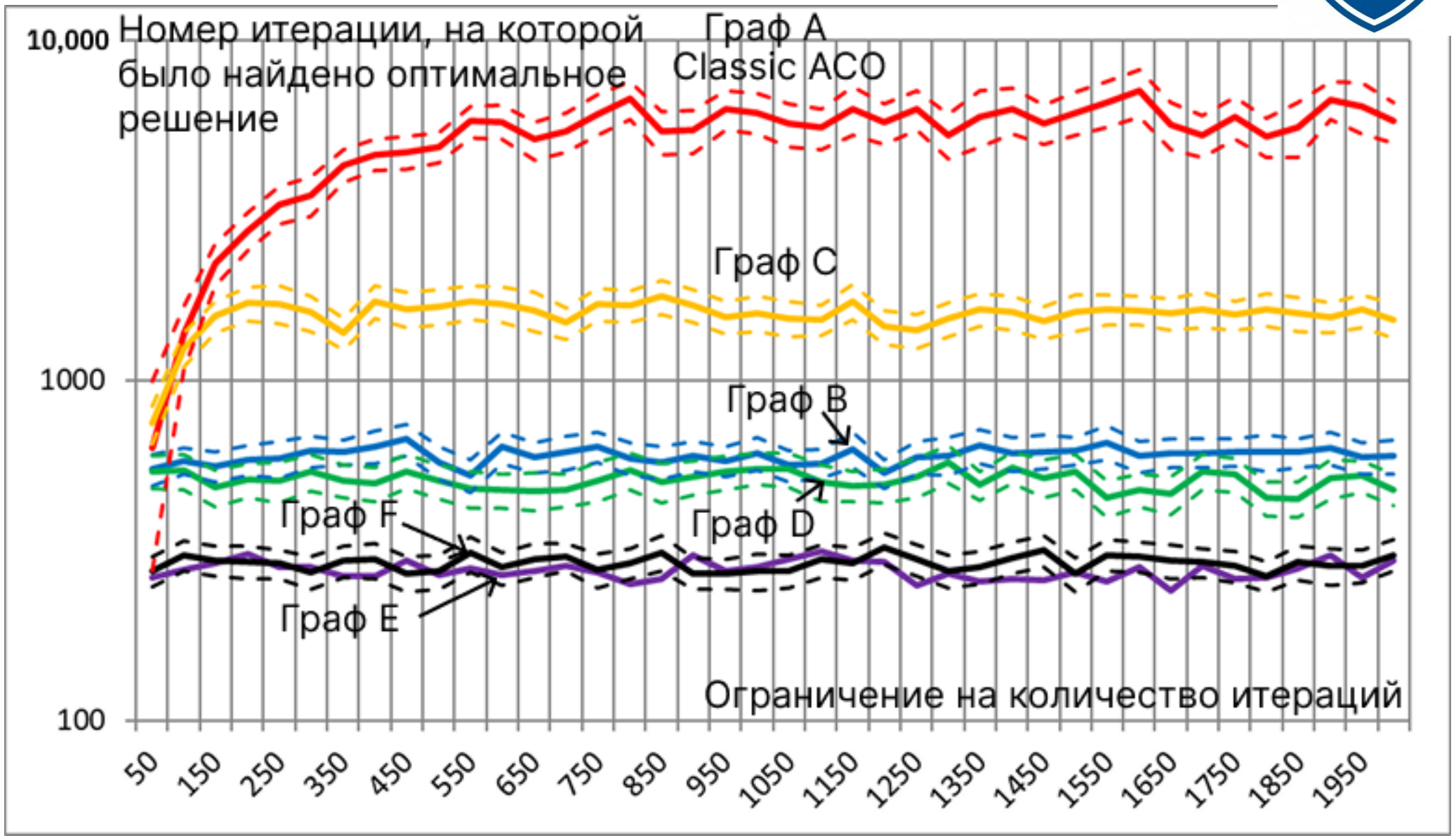


Корневая функция

Функция Растригина

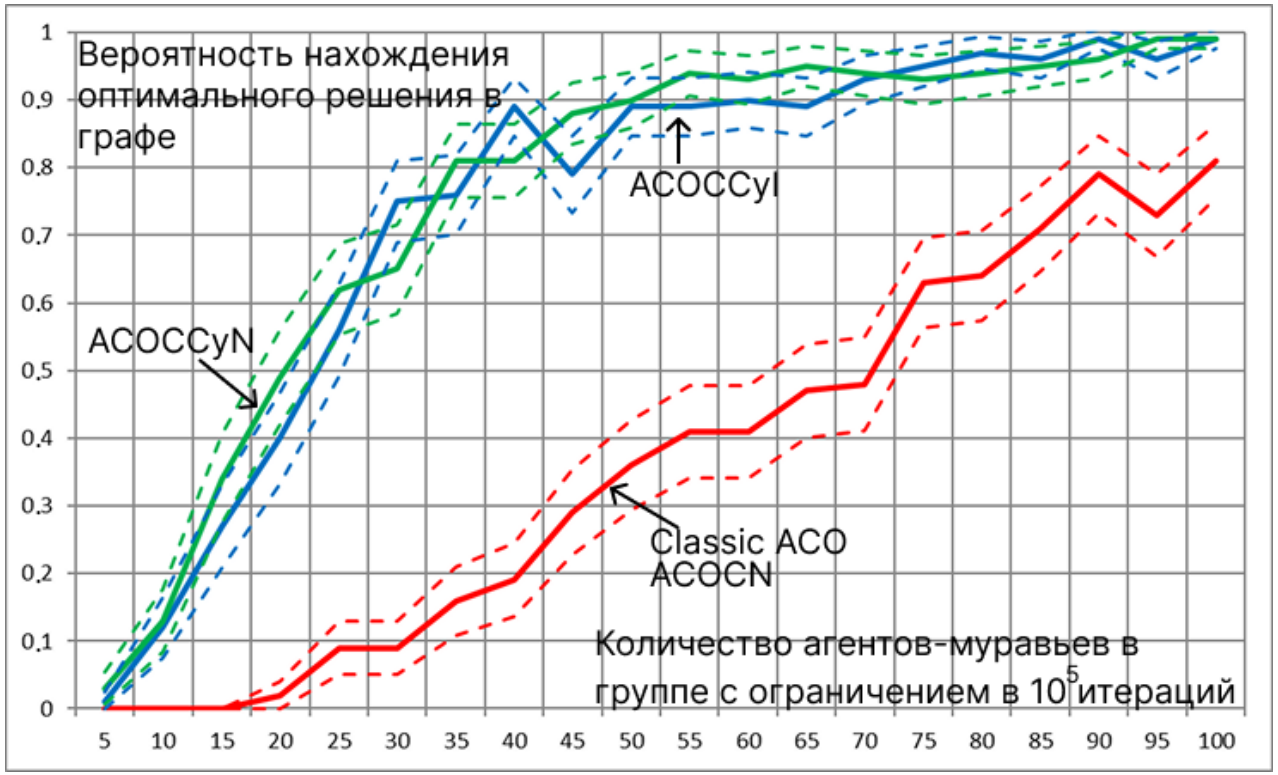


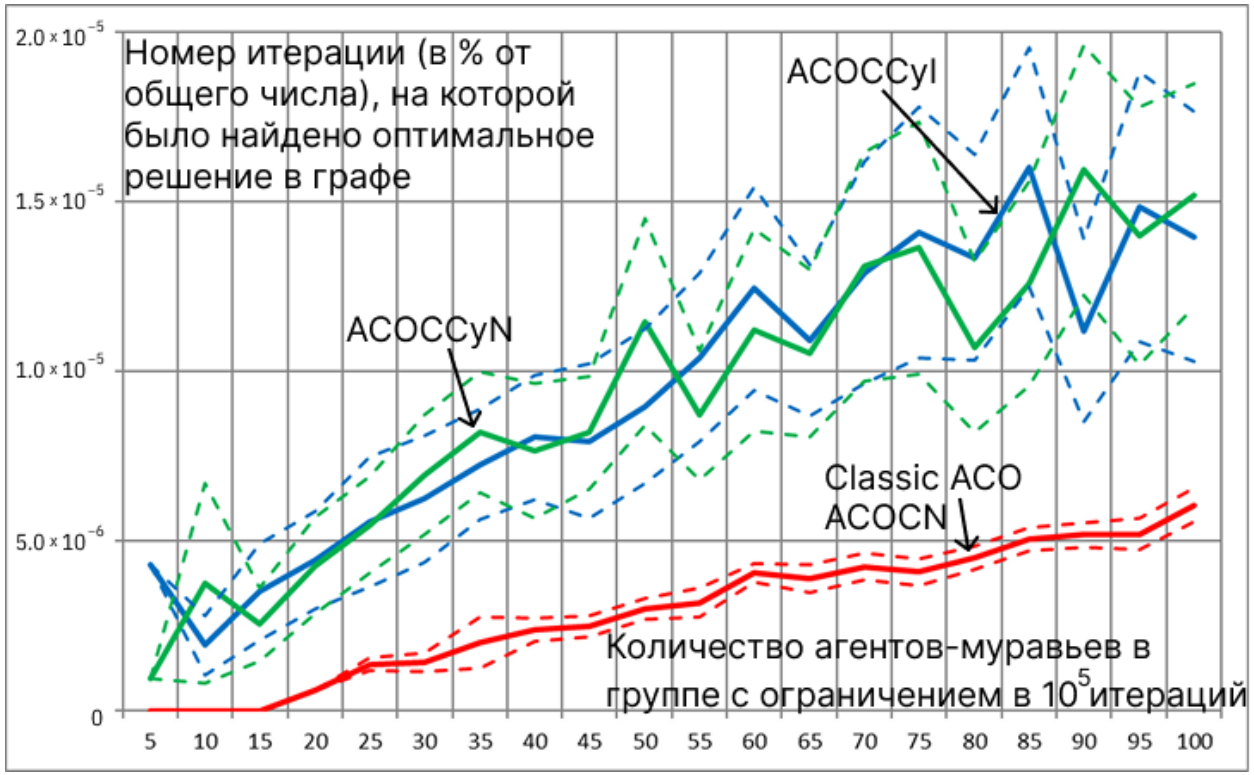
Параметрической граф



Метод муравьиных колоний

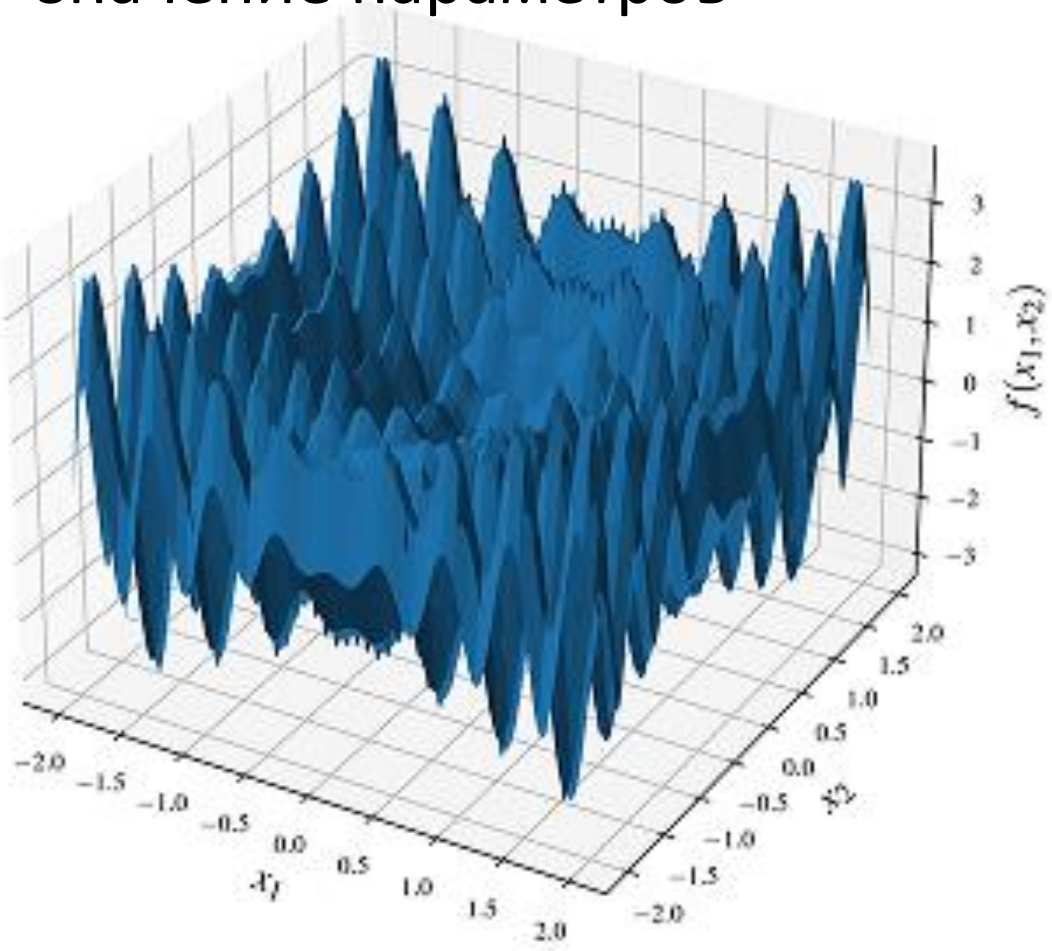
- Исследованы предложенные модификации на различных параметрических графах

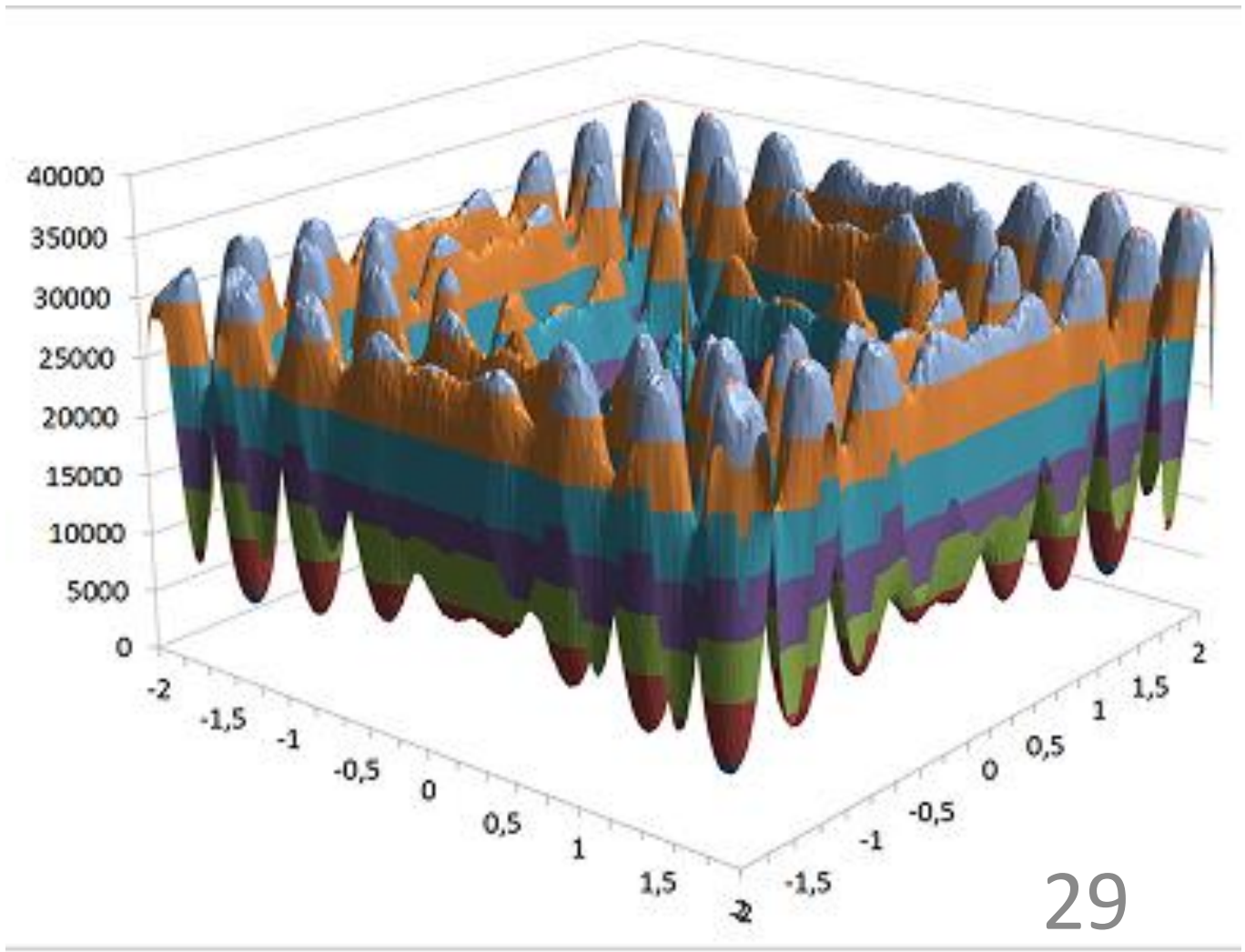




Метод муравьиных колоний

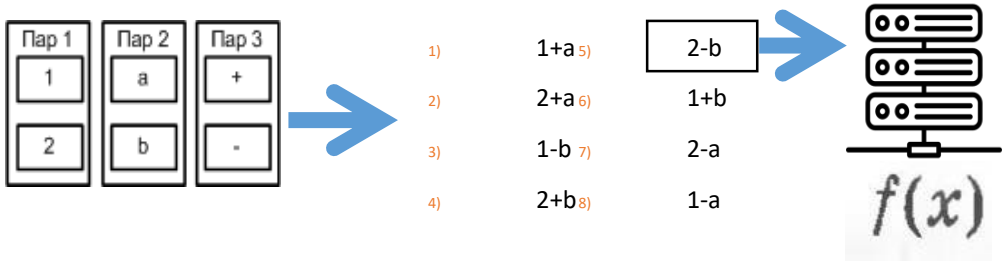
- Исследован номер решения на котором рассмотрено конкретное значение параметров

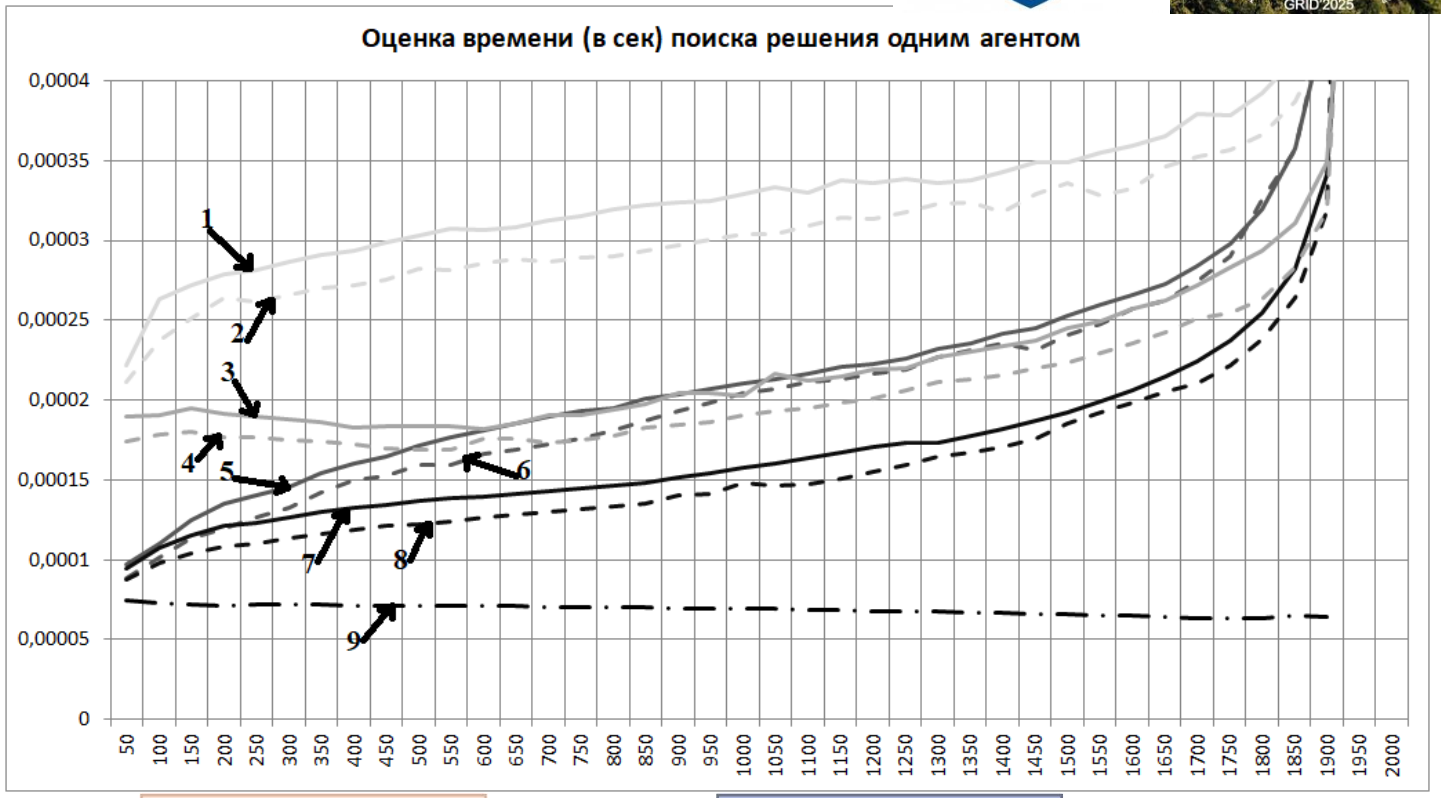


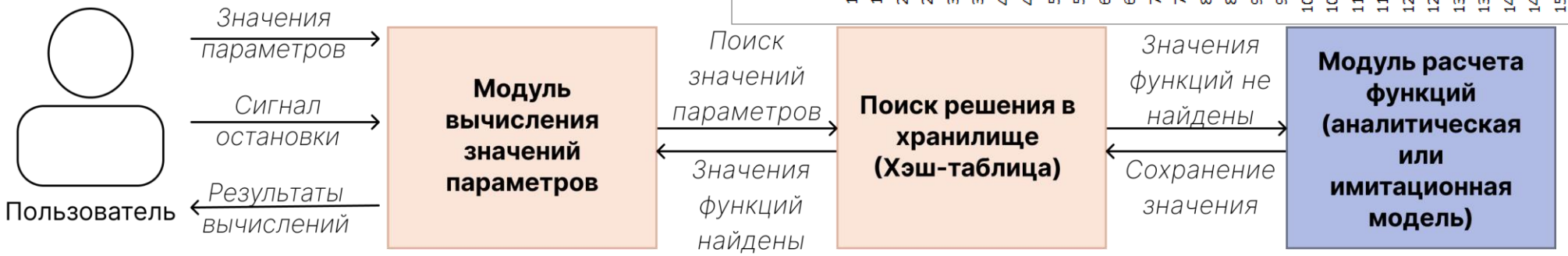


Направленный перебор значений параметров

- Нет строгой функции оптимизации, условия оптимальности определяются пользователем







Параллельный метод муравьиных колоний. Архитектура.

- Агенты перемещаются в worker-thread
- Результаты работы агента хранятся в очереди
- Сокетное соединение с удаленным вычислителем
- Менеджер обращений, управляющий ссылками на граф.
- Графический интерфейс пользователя, взаимодействующий с системой в асинхронном режиме