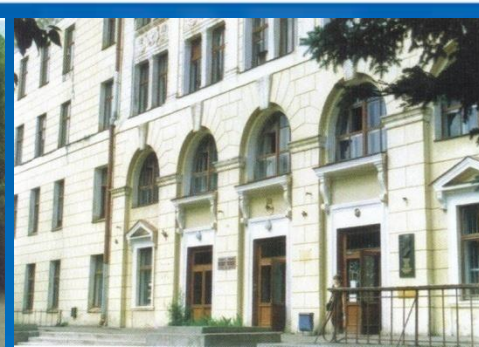
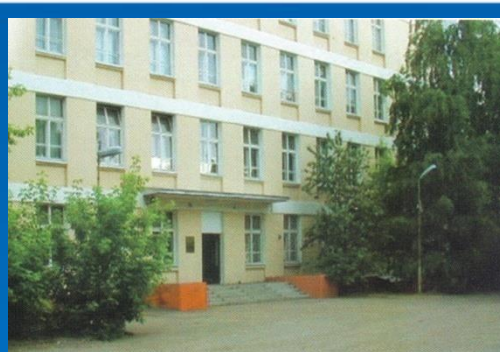
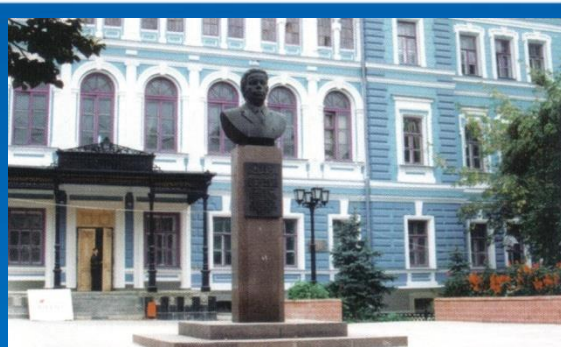




НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. Н.И. ЛОБАЧЕВСКОГО
ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МАТЕМАТИКИ И МЕХАНИКИ





Нижегородский государственный университет им. Н.И. Лобачевского
Институт информационных технологий, математики и механики

Разработка высокопроизводительных математических библиотек для процессоров RISC-V

Суперкомпьютерные дни
в России 2026
Москва, 29 сентября 2026

Волокитин В.Д., Пирова А.Ю.,
Мееров И.Б.

Содержание

Аппаратное обеспечение

Библиотека математических векторных функций

- ☐ Обзор проектов на RVV
- ☐ Разработка RVVMF

Библиотека разреженной алгебры

- ☐ Обзор библиотеки RVVLASparse
- ☐ Результаты тестирования

Аппаратно-программное обеспечение

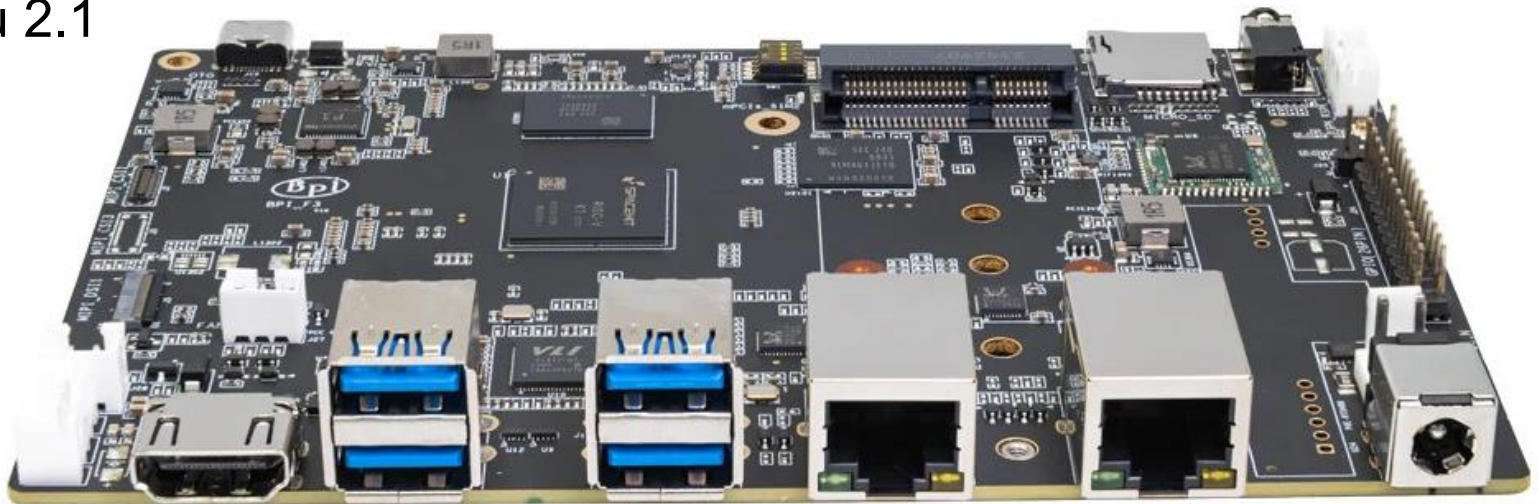
Аппаратное обеспечение:

❑ Banana Pi BPI-F3:

- SpacemiT K1 8x X60(in-order RV64GCVB, RVA22, **RVV 1.0**)
- RVV 1.0: **VLEN = 256** (FPU предположительно 128 бит)
- 4-16 ГБ LPDDR4

Программное обеспечение:

- ▶ GCC 15, Clang 21 – кросс-компиляция
- ▶ OS: Bianbu 2.1



Аппаратно-программное обеспечение

Аппаратное обеспечение:

❑ Banana Pi BPI-SM10:

– SpacemiT K3:

- 8x X100(**Out-of-Order** RV64GCV, RVA23, **RVV 1.0, VLEN = 256 бит**)
- 8x A100(**In-Order** RV64GCV, RVA22, **RVV 1.0, VLEN = 1024 бит**)

– 8 ГБ LPDDR5

❑ Программное обеспечение:

- ▶ GCC 15, Clang 21 – кросс-компиляция
- ▶ OS: Bianbu 4.0.1



Стандарты математических функций

IEEE-754:

- IEEE-754 – международный стандарт, описывающий формат представления чисел с плавающей точкой
 - Полностью описывает работу с FP-числами
 - Используется повсеместно: ПО, компиляторы, аппаратная реализация
 - В редакции 2009 годп появились математические функции:
 - Требование к точности: **0.5 ulp** (всегда возвращается **корректно округленное значение**)

ISO/IEC 10967-2:

- ISO/IEC 10967-2 – международный стандарт, определяющий языконезависимые спецификации для элементарных численных функций
 - Направлен на подробное описание элементарных функций
 - Разделяет функции на базовые и дополнительные
 - Требования по точности менее строгие: базовые до **1 ulp**, дополнительные до **1.5 ulp**

Обзор библиотек. CRLibm

CRLibm:

- ❑ CRLibm – библиотека, обеспечивающая корректное округление во всех точках функции
- ❑ Преимущества:
 - Гарантия точности
 - Поддерживается автоматическая векторизация
 - Есть векторная реализация (x86 - SSE)
- ❑ Недостатки:
 - Из-за проверок точности и алгоритмов уточнения работает **в 2-10 раз** дольше высокопроизводительных библиотек
 - В исходном виде проект больше не поддерживается, его развитие продолжается в виде множества разрозненных форков/библиотек

Обзор библиотек. LibM

LibM:

❑ LibM – стандартная библиотека, которая поставляется вместе с компилятором

❑ Преимущества:

- Общепринятая и кроссплатформенная
- В компиляторе может быть специализированная векторная версия (для RISC-V пока нет)
- Для некоторых версий известны максимальные ошибки (Glibc LibM: https://sourceware.org/glibc/manual/2.41/html_node/Errors-in-Math-Functions.html)
- Есть исследования по точности: Gladman B. et al. Accuracy of mathematical functions in single, double, double extended, and quadruple precision. – 2025.

❑ Недостатки:

- Алгоритмы **плохо поддаются векторизации**
- Часто отсутствует поддержка половинной точности (FP16)

RISC-V 64-bit

expf	1
exp	1
expl	1
expf128	-
exp10f	-
exp10	2
exp10l	2
exp10f128	-

Обзор библиотек. `veclibm`

`veclibm`:

❑ `veclibm` – библиотека, написанная для RISC-V Vector Extension

❑ Преимущества:

- Высокопроизводительная
- Реализована с использованием RVV Intrinsics
- Достаточно точная (заявляется $< 1 \text{ ulp}$, хотя мы находили функции $> 1 \text{ ulp}$)

❑ Недостатки:

- Реализовано **только для *double***
- Использует интерфейс для работы с массивами с оптимально подобранным параметром LMUL
- Согласно документации, тестирование проводится в эмуляторе (QEMU), а не на реальном оборудовании
- Проект давно не обновлялся

Обзор библиотек. Sleef

Sleef:

- ❑ Sleef – кроссплатформенная векторная библиотека математических функций
- ❑ Преимущества:
 - Универсальная кроссплатформенная библиотека, написанная на псевдоинтринсиках
- ❑ Недостатки:
 - Реализовано только для LMUL=1,2
 - Реальная точность вычислений часто **не соответствует заявленной**
 - Обязательно требует -ffp-contract=off (отключение FMA / стягивания операций), что может снизить производительность

RVVMF

RVVMF:

❑ RVVMF – векторная библиотека математических функций под RVV

❑ Цель:

- Высокопроизводительная библиотека
- Точность удовлетворяет и превосходит требования ISO/IEC 10967-2 (**0.501 ulp** статистически)

❑ Что уже готово:

- Широкий набор функций: `exp`, `exp2`, `expm1`, `log`, `log2`, `log10`, `log1p`, `sin`, `cos`, `tan`, `atan`, `atan2`, `sinh`, `cosh`, `tanh`, `sqrt`, `pow`, `cbrt`, `hypot`
- Тестовая система:
 - Проверка точности, флагов `FE_EXCEPTION` и монотонности
 - Оценка производительности (задержка и пропускная способность)

❑ Планы на будущее:

- Расширение списка функций и оптимизация
- Поддержка вычислений со сниженной точностью (для повышения производительности)

*Проект библиотеки RVVMF поддержан ООО «КНС ГРУПП»

RVVMF. Основные проблемы и решения

RVVMF:

❑ Проблема: потеря точности вычислений

- Решение: использование **doubled-double арифметики** (представление числа или константы в виде суммы двух чисел соответствующей точности):
 - $c = a * b; \Rightarrow c_h = a * b; c_l = fma(a, b, -c);$

❑ Флаги **FE_EXCEPTION**:

- FE_INEXACT: контроль ослаблен, так как стандарт допускает установку флага при многих операциях
- FE_DIVBYZERO: встречается только в особых функциях и особых точках (например, $pow(0.0, -2)$)
- FE_INVALID, FE_UNDERFLOW и FE_OVERFLOW: строгий контроль во всех особых точках и областях

❑ Производительность:

- Сложности профилирования кода на целевой архитектуре -> Решено **ручной оптимизацией**

RVVMF. Результаты

	exp	exp2	expm1	log	log2	log10	log1p	sin	cos	tan	atan	atan2
glibc FP64	0.008	0.006	0.443	0.005	0.010	2.234*	0.554	0.699	0.690	1.230	0.006	0.008
veclibm FP64	0.119	0.121	0.377	0.032	0.013	0.021	0.130	9.091	8.933	2.850	0.299*	0.603*
SLEEF FP64	0.380	0.498	0.178	0.164	0.145	0.138	0.602	8.797	9.663	6.030	0.364	0.254
<u>RVVMF FP64</u>	<u>0.006</u>	<u>0.013</u>	<u>0.280</u>	<u>0.003</u>	<u>0.003</u>	<u>0.003</u>	<u>0.614</u>	<u>0.891</u>	<u>0.949</u>	<u>0.713</u>	<u>0.001</u>	<u>0.001</u>
glibc FP32	0.031	0.037	2.927	0.216	0.140	13.407*	3.655	6.936	6.570	18.622	4.942	224.034
SLEEF FP32	3.961	2.621	1.555	0.383	0.256	0.329	3.341*	26.188	30.338	63.432	0.967	1.359
<u>RVVMF FP32</u>	<u>0.003</u>	<u>0.009</u>	<u>0.049</u>	<u>0.014</u>	<u>0.010</u>	<u>0.010</u>	<u>0.100</u>	<u>0.577</u>	<u>0.590</u>	<u>0.054</u>	<u>0.457</u>	<u>0.011</u>
<u>RVVMF FP16</u>	<u>0.110</u>	<u>0.110</u>	<u>0.646</u>	<u>0.473</u>	<u>0.378</u>	<u>0.473</u>	<u>0.892</u>	<u>0.788</u>	<u>0.567</u>	<u>0.851</u>	<u>0.378</u>	<u>0.589</u>

	hypot	sinh	cosh	tanh	sqrt	pow	cbrt
glibc FP64	3.316	3.780	0.611	5.099*	0	0.008	323.6*
veclibm FP64	-	0.118*	0.125	0.203	-	0.078	2.594
SLEEF FP64	0.001	0.061**	0.002**	0.073*	0	0.114	0.279
<u>RVVMF FP64</u>	<u>3.490</u>	<u>0.483</u>	<u>0.012</u>	<u>0.088</u>	<u>0</u>	<u>0.012</u>	<u>0</u>
glibc FP32	0	16.039	4.152	18.976*	0	0.039	106.840
SLEEF FP32	0.063	0.673**	0.075**	0.243*	0	0.784	0
<u>RVVMF FP32</u>	<u>12.667</u>	<u>0.004</u>	<u>0.159</u>	<u>0.187</u>	<u>0</u>	<u>0.161</u>	<u>0</u>
<u>RVVMF FP16</u>	<u>47.301</u>	<u>0.126</u>	<u>0.032</u>	<u>0.347</u>	<u>0</u>	<u>0.106</u>	<u>261.2*</u>

Количество точек с неверным округлением на 1000 случайных:

* — есть точки >1 ULP

** — проблемы с областью

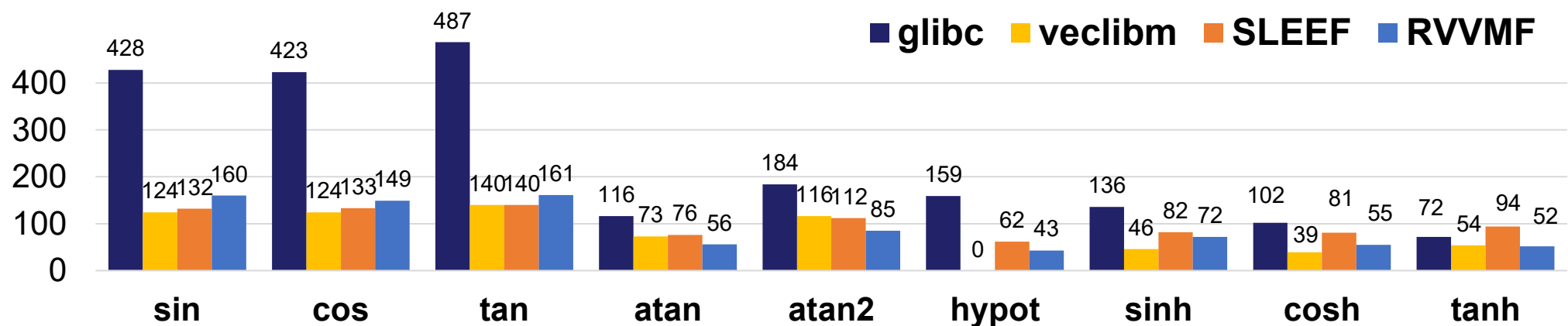
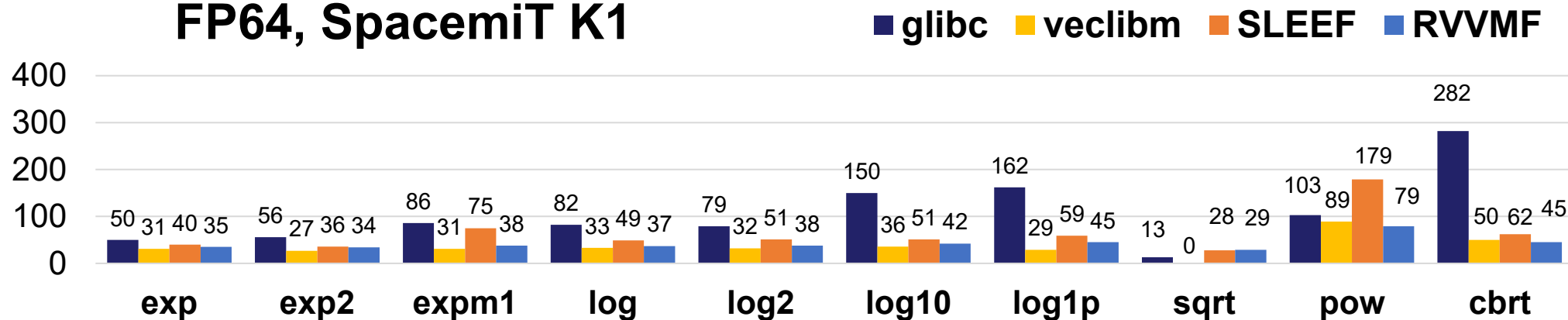
определения

 — соответствует **0.501 ULP**

 — не соответствует **0.501 ULP**

RVVMF. Производительность K1

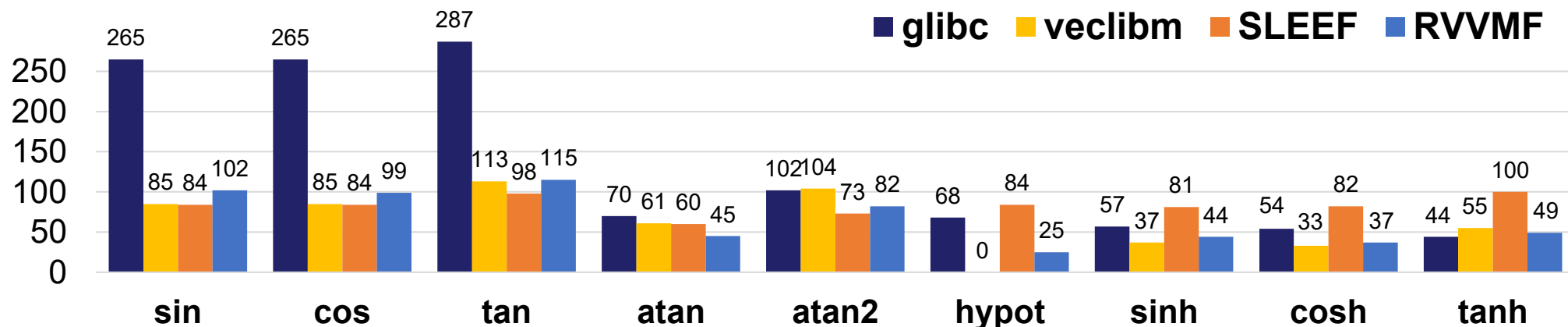
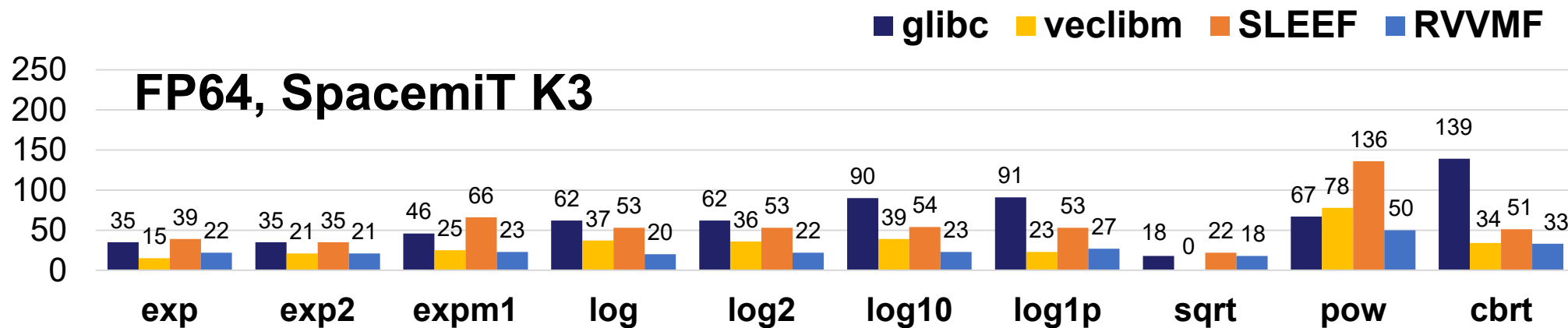
FP64, SpacemiT K1



❑ Производительность на уровне конкурентов при более высокой точности

❑ В ряде случаев наблюдается значительный **выигрыш** в скорости

RVVMF. Производительность K3



☐ Ускорение работы на новой архитектуре K3 для всех библиотек

Разреженная алгебра на RISC-V

- ❑ Операции с **разреженными матрицами** – ключевые элементы программных пакетов для научного и инженерного моделирования
- ❑ Умножение разреженной матрицы на вектор (**SpMV**) – вычислительное ядро итерационных решателей СЛАУ.

$$y := \alpha Ax + \beta y$$

$A \in \mathbb{R}^{n \times m}$ – разреженная матрица, $x \in \mathbb{R}^m$, $y \in \mathbb{R}^n$ – плотные векторы, $\alpha, \beta \in \mathbb{R}$ – вещественные скаляры.

❑ Проблема:

- производительность SpMV сильно зависит от формата хранения матрицы и алгоритма умножения. Заранее неизвестно, какой формат окажется оптимальным для конкретной матрицы.

Разреженная алгебра на RISC-V

□ RISC-V vs x86:

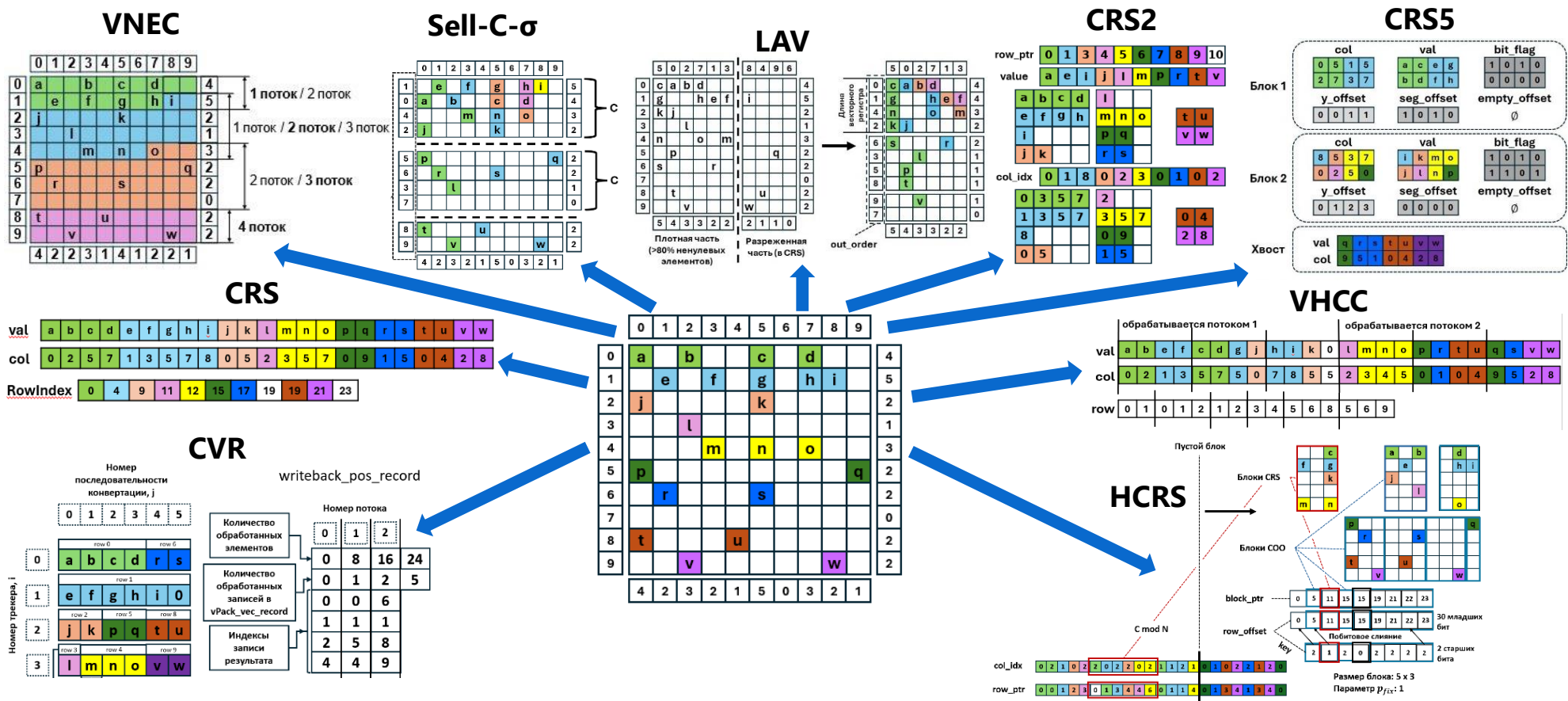
x86	RISC-V
Доступны вендорные и открытые реализации Sparse BLAS	Отсутствует эталонная реализация Sparse BLAS, в сторонних библиотеках — лишь частичная поддержка RVV
Разработано множество форматов хранения, оптимизированных под SIMD-инструкции	Опубликованы векторные версии лишь для отдельных форматов
Реализации нестандартных форматов представлены в исследовательских проектах (часто с открытым исходным кодом)	Исходный код нестандартных форматов, как правило, недоступен

RVVLA Sparse

- ❑ **RVVLA Sparse** – открытая библиотека операций с разреженными матрицами для процессоров RISC-V, реализованная с использованием интринсиков RVV 1.0 и технологии OpenMP.
 - Поддерживается операция **SpMV** для **9 форматов** хранения: CSR, Sell-C-σ, CSR2, CSR5, CVR, VHCC, VNEC, LAV, собственный формат **HCSR**.
 - **В разработке**: операции SpMM, маскированный SpMM, TRSM, TRSV.
- ❑ Реализация доступна:
<https://github.com/UNN-ITMM-Software/RVVLA Sparse>
- ❑ Дополнительно: ML-модель для автоматического подбора формата по портрету матрицы

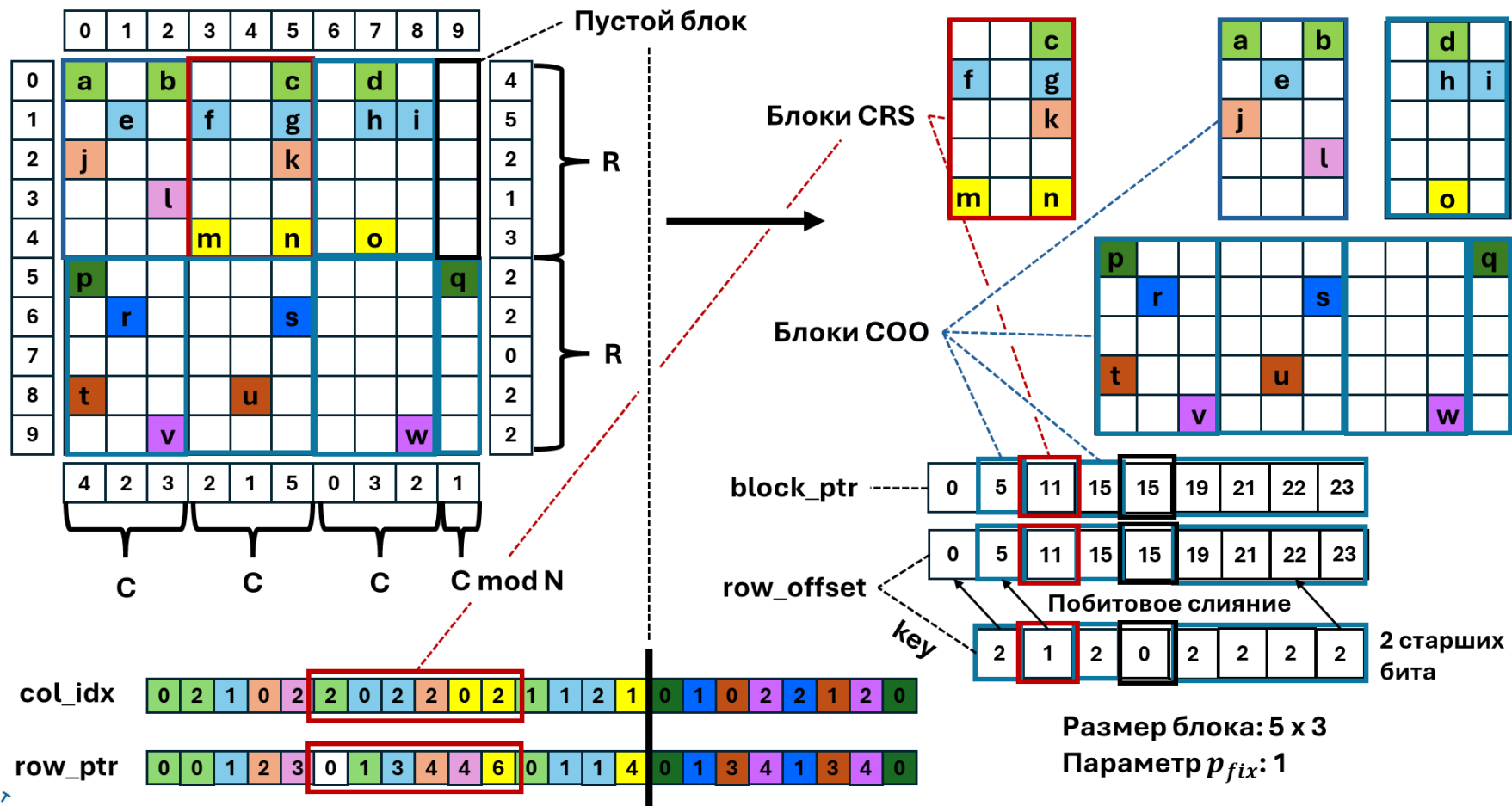
* Работа выполнена при поддержке программы академического лидерства ННГУ "Приоритет-2030"

Форматы хранения разреженной матрицы



Новый формат – Hierarchical CSR

- Идея: улучшение пространственной локальности обращений к памяти при работе с вектором x

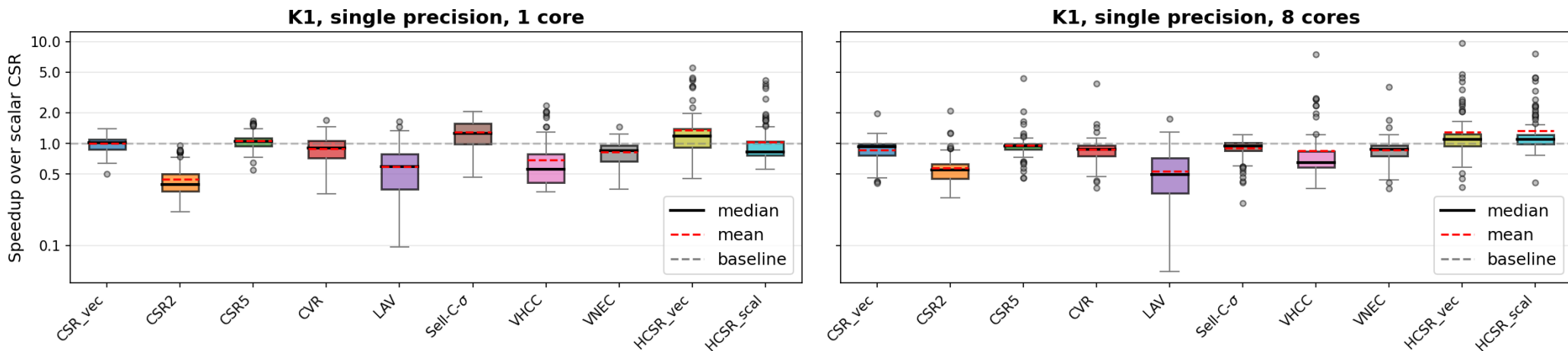


Тестирование

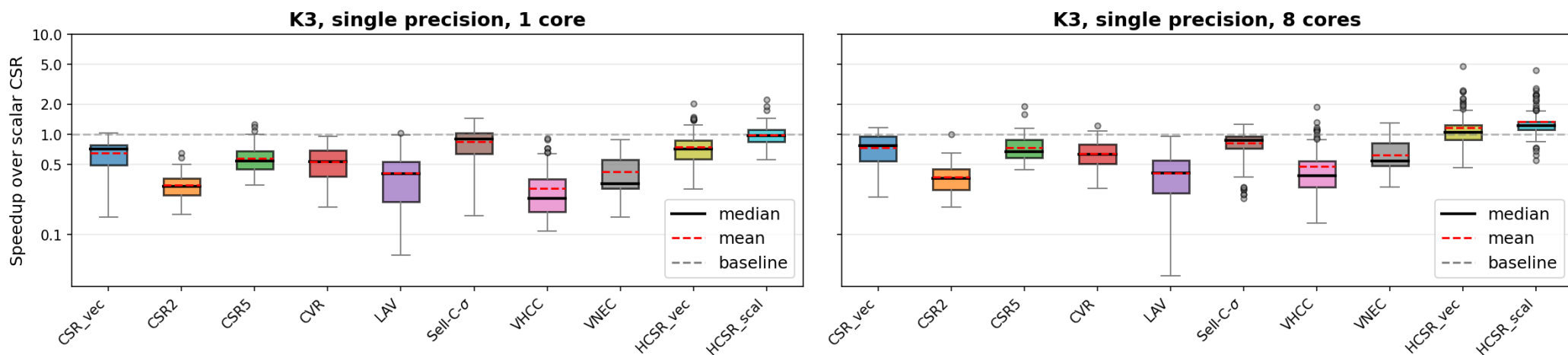
- ❑ Операция **SpMV**: $y := \alpha Ax + \beta y$
- ❑ Тестовая выборка: 121 матрица из коллекции SuiteSparse размерами от $0.5 \cdot 10^3 \times 400 \cdot 10^3$ до $8.39 \cdot 10^6 \times 8.39 \cdot 10^6$ и заполненностью от $3.1 \cdot 10^{-7}$ до $1.4 \cdot 10^{-2}$.
Плотный векторы заполнены случайными числами из $[0, 1)$, значения $\alpha = 1.0$, $\beta = 0.0$ фиксированы
- ❑ Метрика – ускорение SpMV в данном формате относительно реализации для **скалярного CSR**

Вычислительные эксперименты.

Ускорение SpMV относительно скалярного CSR



SpacemiT K1: лучшие – Sell-C- σ на 1 ядре, HCSR на 8 ядрах



SpacemiT K3: лучшие – скалярный CSR на 1 ядре, HCSR на 8 ядрах

Вычислительные эксперименты.

Ускорение SpMV относительно скалярного CSR

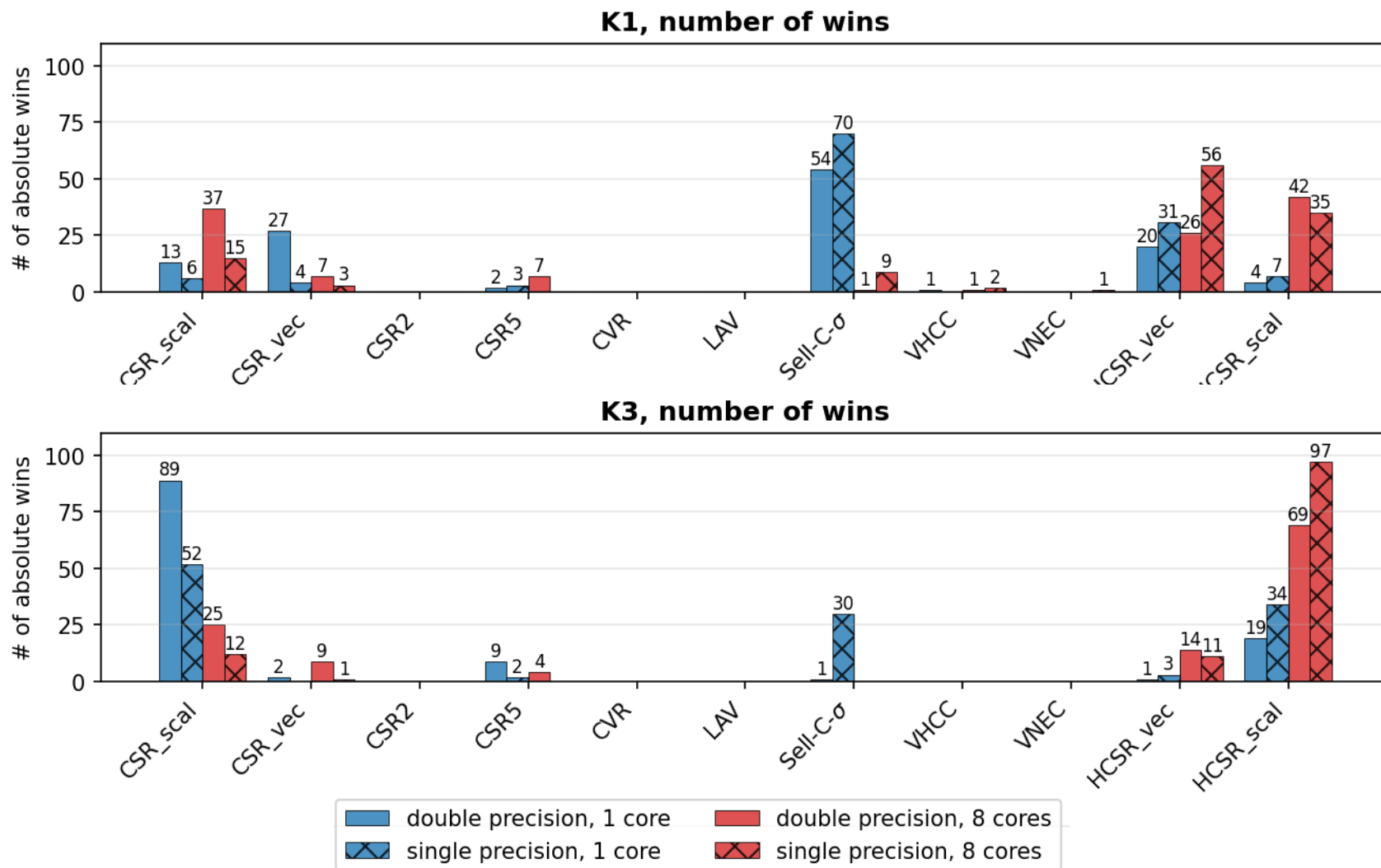
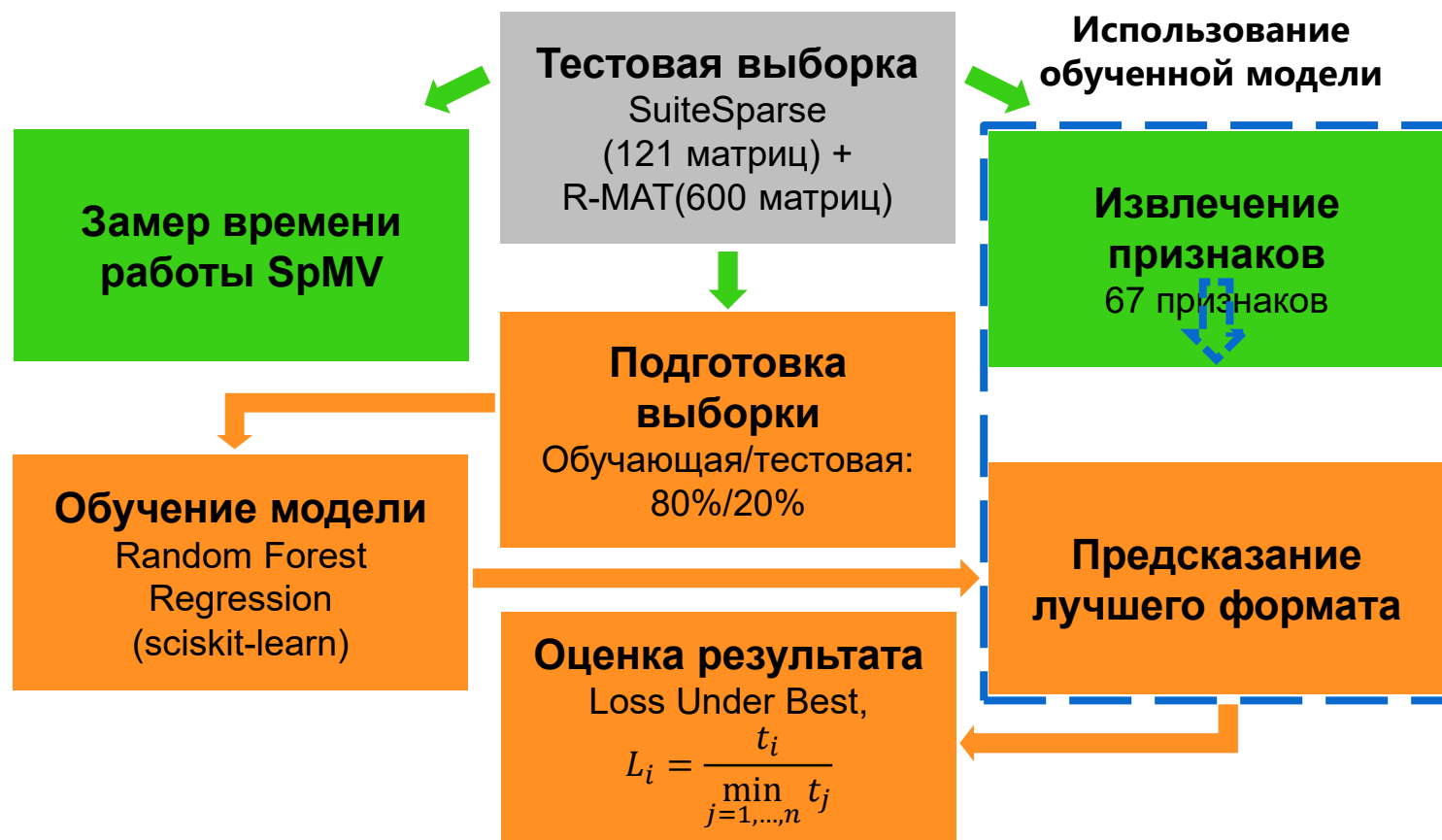
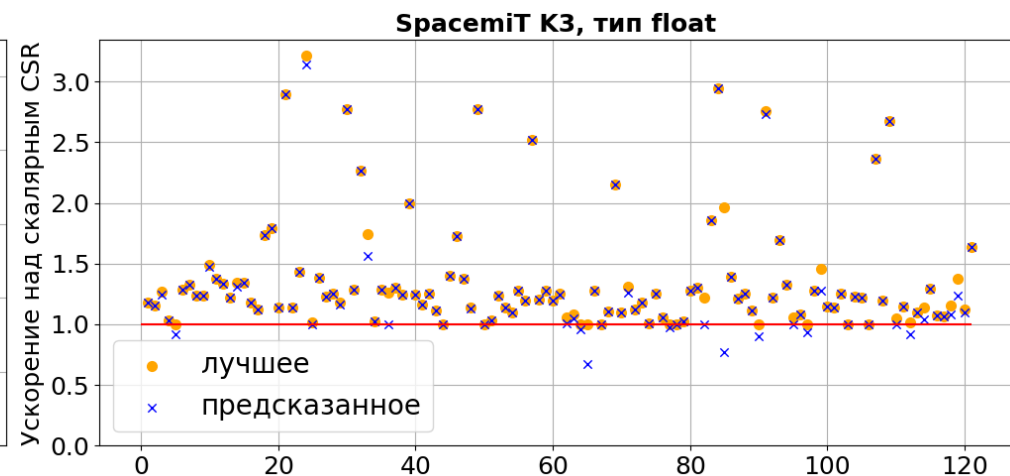
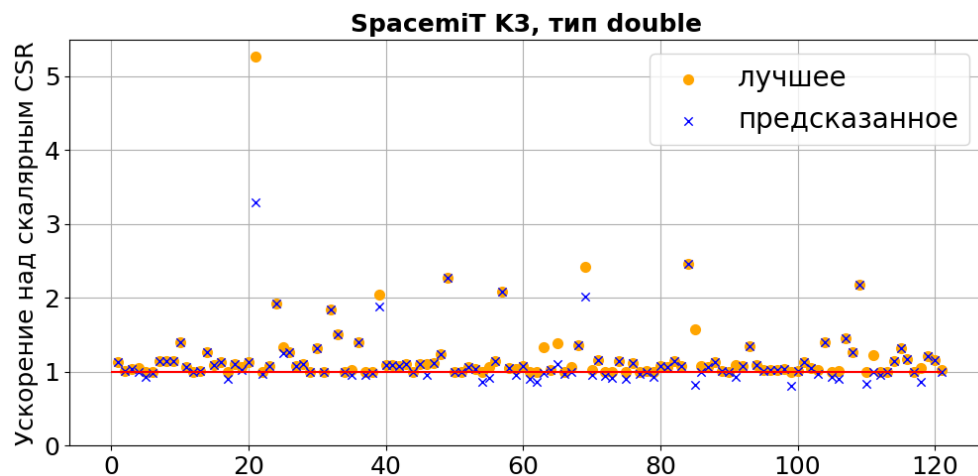
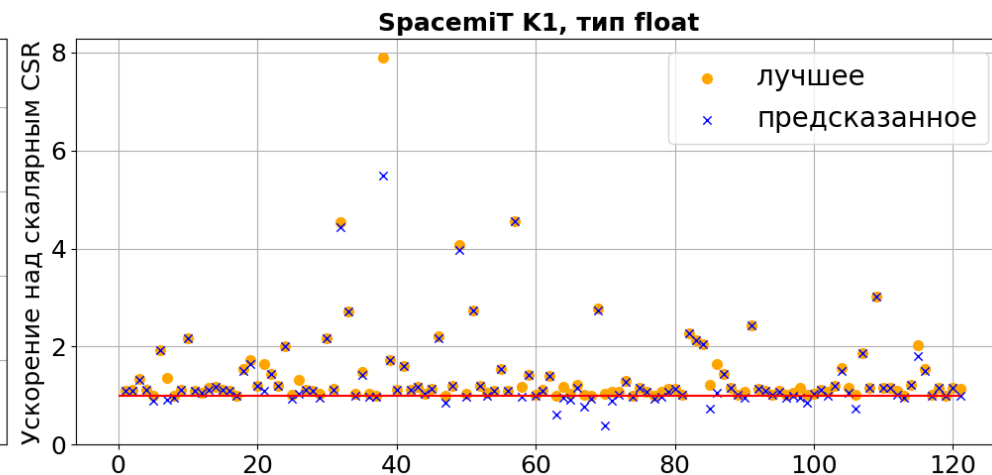
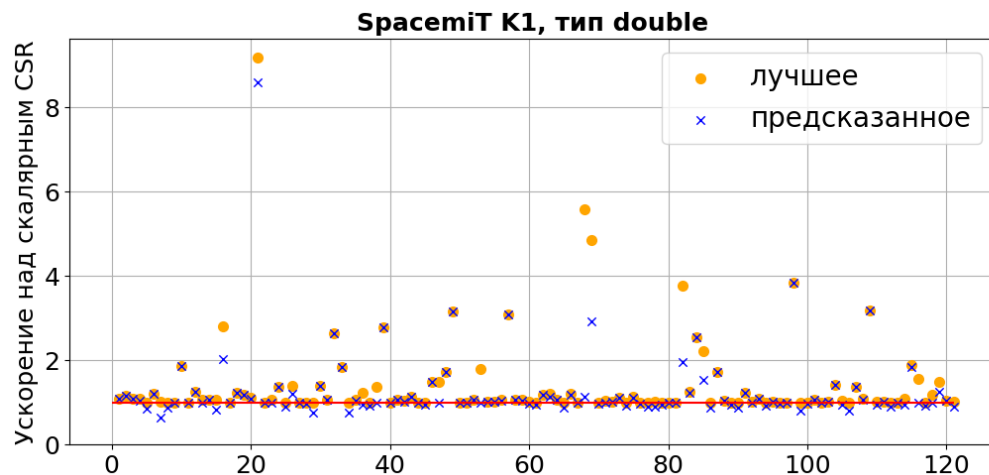


Схема обучения и тестирования модели автоматического подбора формата

- **Цель:** определить наиболее эффективный формат матрицы



Результаты тестирования модели автоматического подбора формата



ID матрицы

Анализ результатов SpMV

- ❑ Лучшие результаты получены в форматах **CSR**, **Sell-C-Sigma**, векторном и скалярном **HCSR**.
- ❑ На SparceMiT K1 векторизация и параллелизм ускоряют вычисления.
 - Среднее ускорение HCSR – 1,15x (double) и 1,35x (float) на 1 ядре; 1,11x и 1,30x на 8 ядрах.
- ❑ На SparceMiT K3 выигрыш от векторизации есть только в многопоточной версии.
 - Среднее ускорение скалярного HCSR на 8 ядрах: 1,12x (double) и 1,34x (float).

Общие выводы

□ Преимущества:

- Современные RISC-V платы имеют хороший потенциал для исследования векторизации в вычислительных задачах
- Наблюдается **существенное** и **планомерное развитие** от поколения к поколению процессоров

□ Недостатки:

- Неизвестна эффективность аппаратной реализации специфичных инструкций. Судя по полученным результатам, индексная загрузка «скатывается» к скалярной загрузке из памяти
- Широко доступны только **малые одноплатные компьютеры**, серверные решения существуют в единичных экземплярах