ИССЛЕДОВАНИЕ ЭФФЕКТИВНОСТИ ВЕКТОРИЗАЦИИ ГНЕЗД ЦИКЛОВ С НЕРЕГУЛЯРНЫМ ЧИСЛОМ ИТЕРАЦИЙ

Рыбаков Алексей Анатольевич

Шумилин Сергей Сергеевич

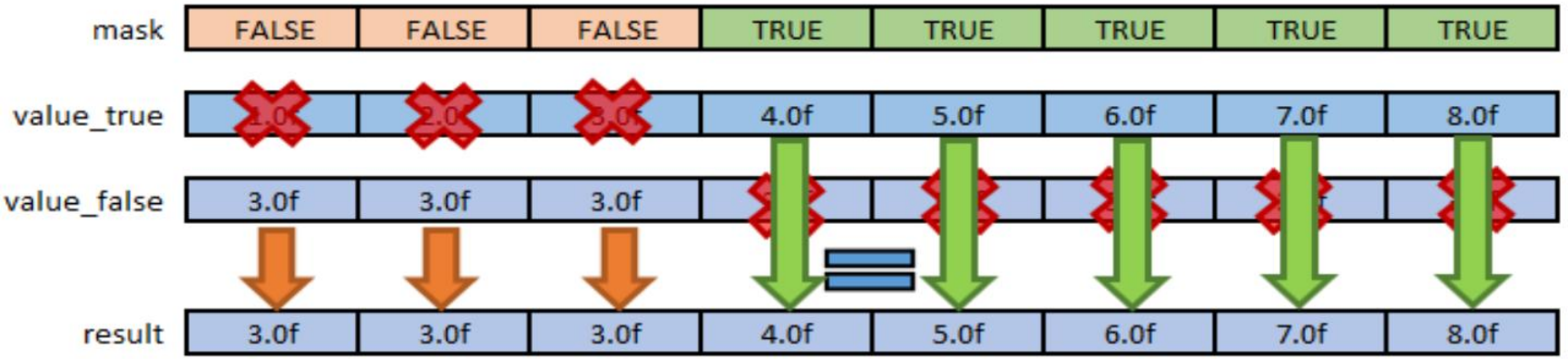
НСКФ 27-30 НОЯБРЯ 2018
СЕДЬМОЙ НАЦИОНАЛЬНЫЙ СУПЕРКОМПЬЮТЕРНЫЙ ФОРУМ

АНО "НСКФ"

INTEL® ADVANCED VECTOR EXTENSIONS 512

IN INTEL® XEON® SCALABLE PROCESSORS

Skylake — Knights Landing — Broadwell
Comparative analysis of vector processing functionality



взято с сайта tech.io

Пример применения **масок** в векторных операциях

intel Intrinsics Guide

_mm_search ?

Technologies

- [ ] MMX
- [ ] SSE
- [ ] SSE2
- [ ] SSE3
- [ ] SSSE3
- [ ] SSE4.1
- [ ] SSE4.2
- [ ] AVX
- [ ] AVX2
- [ ] FMA
- [x] AVX-512
- [ ] KNC
- [ ] SVML
- [ ] Other

__m512i _mm512_4dpwssd_epi32 (_m512i src, _m512i a0, _m512i a1, _m512i a2, _m512i a3, _m128i * b)	vp4dpwssd
__m512i _mm512_mask_4dpwssd_epi32 (_m512i src, _mmask16 k, _m512i a0, _m512i a1, _m512i a2, _m512i a3, _m128i * b)	vp4dpwssd
__m512i _mm512_maskz_4dpwssd_epi32 (_mmask16 k, _m512i src, _m512i a0, _m512i a1, _m512i a2, _m512i a3, _m128i * b)	vp4dpwssd
__m512i _mm512_4dpwssds_epi32 (_m512i src, _m512i a0, _m512i a1, _m512i a2, _m512i a3, _m128i * b)	vp4dpwssds
__m512i _mm512_mask_4dpwssds_epi32 (_m512i src, _mmask16 k, _m512i a0, _m512i a1, _m512i a2, _m512i a3, _m128i * b)	vp4dpwssds
__m512i _mm512_maskz_4dpwssds_epi32 (_m512i src, _mmask16 k, _m512i a0, _m512i a1, _m512i a2, _m512i a3, _m128i * b)	vp4dpwssds
__m512 _mm512_4fmadd_ps (_m512 a, _m512i b0, _m512i b1, _m512i b2, _m512i b3, _m128i * c)	v4fmaddps
__m512 _mm512_mask_4fmadd_ps (_m512 a, _mmask16 k, _m512i b0, _m512i b1, _m512i b2, _m512i b3, _m128i * c)	v4fmaddps
__m512 _mm512_maskz_4fmadd_ps (_m512 a, _mmask16 k, _m512i b0, _m512i b1, _m512i b2, _m512i b3, _m128i * c)	v4fmaddps

Synopsis

```
__m512i _mm512_mask_add_epi32 (__m512i src, __mmask16 k, __m512i a, __m512i b)
#include <immintrin.h>
Instruction: vpaddd zmm {k}, zmm, zmm
CPUID Flags: AVX512F for AVX-512, KNCNI for KNC
```

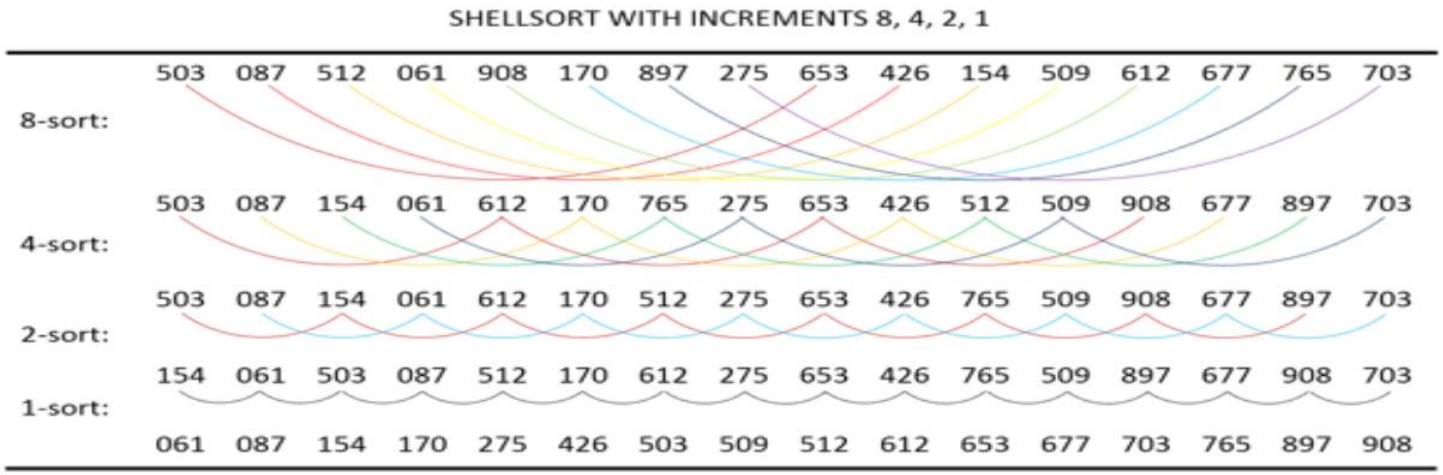
Description

Add packed 32-bit integers in a and b, and store the results in dst using writemask k (elements are copied from src when the corresponding mask bit is not set).

Operation

```
FOR j := 0 to 15
        i := j*32
        IF k[j]
                dst[i+31:i] := a[i+31:i] + b[i+31:i]
        ELSE
                dst[i+31:i] := src[i+31:i]
        FI
ENDFOR
dst[MAX:512] := 0
```

взято с сайта software.intel.com



Визуализация сортировки Шелла

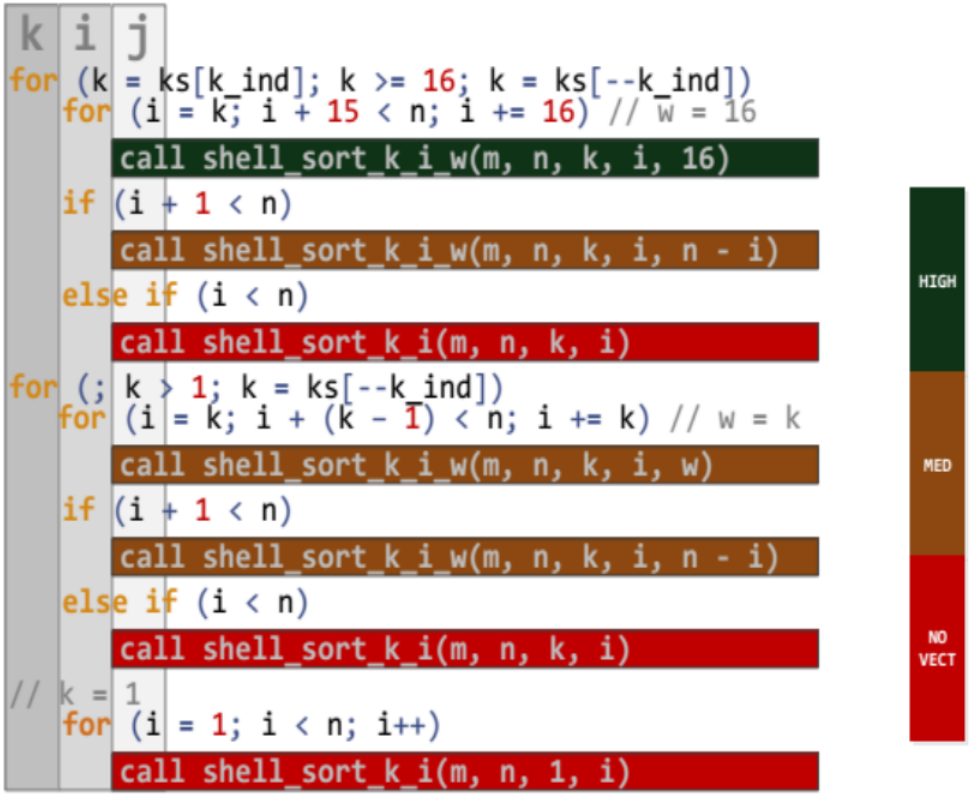
Взято с GitHub, польз. heray1990

Последовательность	**Формула**
Последовательность Шелла, 1959 г.	$k_1 = \lfloor \frac{N}{2} \rfloor, k_i = \lfloor \frac{k_{i-1}}{2} \rfloor, k_t = 1$
Последовательность Хиббарда, 1963 г.	$2^i - 1 \leq N, i \in \mathbb{N}$
Последовательность Пратта, 1971 г.	$2^i \cdot 3^j \leq \frac{N}{2}, i \in \mathbb{N}, j \in \mathbb{N}$
Последовательность Седжвика, 1986 г.	$k_i = \begin{cases} 9 \cdot 2^i - 9 \cdot 2^{\frac{i}{2}} + 1, k\ even \\ 8 \cdot 2^i - 6 \cdot 2^{\frac{i+1}{2}} + 1, k\ odd \end{cases}$

Последовательности шагов,
используемые в сортировке Шелла

```
01 void shell_sort(float *m, int n, int *ks, int k_ind)
02 {
03     int i, j, k;
04
05     for (k = ks[k_ind]; k > 0; k = ks[--k_ind])
06     {
07         for (i = k; i < n; i++)
08         {
09             float t = m[i];
10
11             for (j = i; j >= k; j -= k)
12             {
13                 if (t < m[j - k])
14                 {
15                     m[j] = m[j - k];
16                 }
17                 else
18                 {
19                     break;
20                 }
21             }
22
23             m[j] = t;
24         }
25     }
26 }
```

Каноничная реализация сортировки Шелла



Декомпозиция сортировки Шелла для выделения **векторизуемых участков кода**

```
int j = i;
float t = m[j];

do
{
    bool p1 = (j >= k);

    if (!p1)
    {
        break;
    }

    float q = m[j - k];
    bool p2 = (t < q);

    if (!p2)
    {
        break;
    }

    m[j] = q;
    j -= k;
}
while (true);

m[j] = t;
```

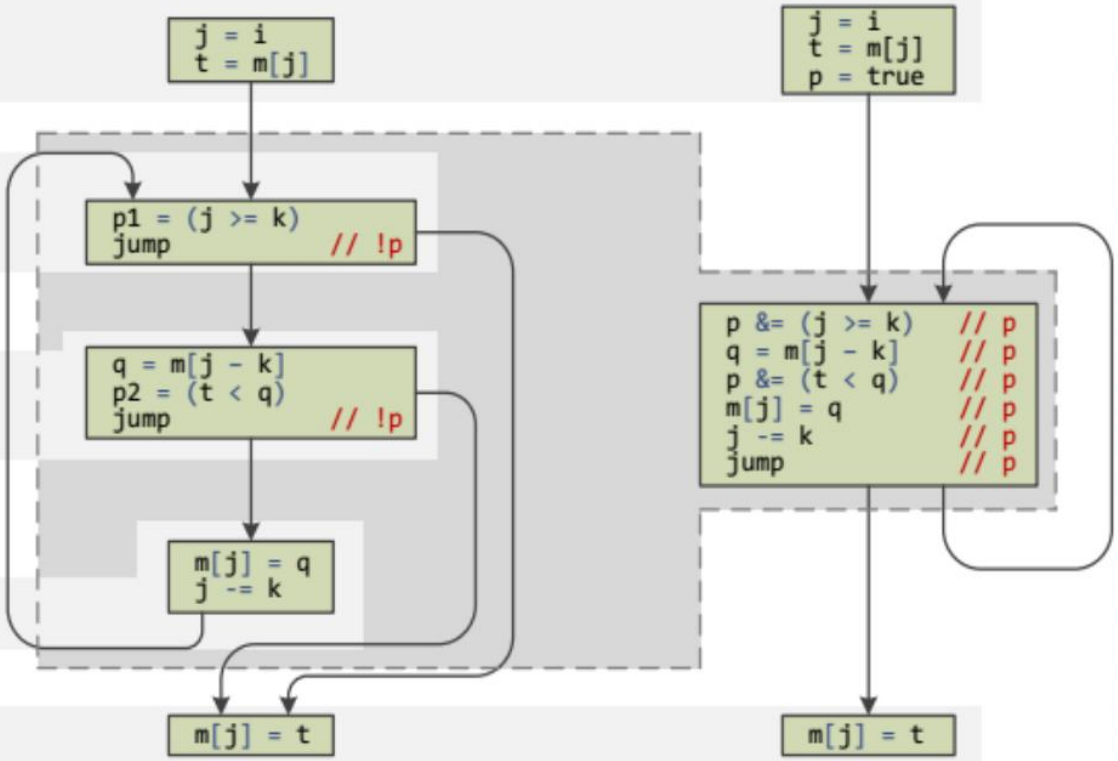


Схема перевода **тела внутреннего цикла** сортировки Шелла в **предикатную форму**

```
01 void shell_sort_k_i_w(float *m, int n, int k, int i, int w)
02 {
03     int j = i;
04     __mmask16 ini_mask = ((unsigned int)0xFFFF) >> (16 - w);
05     __mmask16 mask = ini_mask;
06     __m512i ind_j = _mm512_add_epi32(_mm512_set1_epi32(i),
07                                      ind_straight);
08     __m512 t, q;
09
10     t = _mm512_mask_load_ps(t, mask, &m[j]);
11
12     do
13     {
14         mask = mask & _mm512_mask_cmp_epi32_mask(mask, ind_j, ind_k,
15                                                  _MM_CMPINT_GE);
16         q = _mm512_mask_load_ps(q, mask, &m[j - k]);
17         mask = mask & _mm512_mask_cmp_ps_mask(mask, t, q,
18                                               _MM_CMPINT_LT);
19         _mm512_mask_store_ps(&m[j], mask, q);
20         ind_j = _mm512_mask_sub_epi32(ind_j, mask, ind_j, ind_k);
21         j -= k;
22     }
23     while (mask != 0x0);
24
25     _mm512_mask_i32scatter_ps(m, ini_mask, ind_j, t, _MM_SCALE_4);
26 }
```

Векторизованный вариант ядра
сортировки Шелла

_mm512_add_epi32

_mm512_set1_epi32

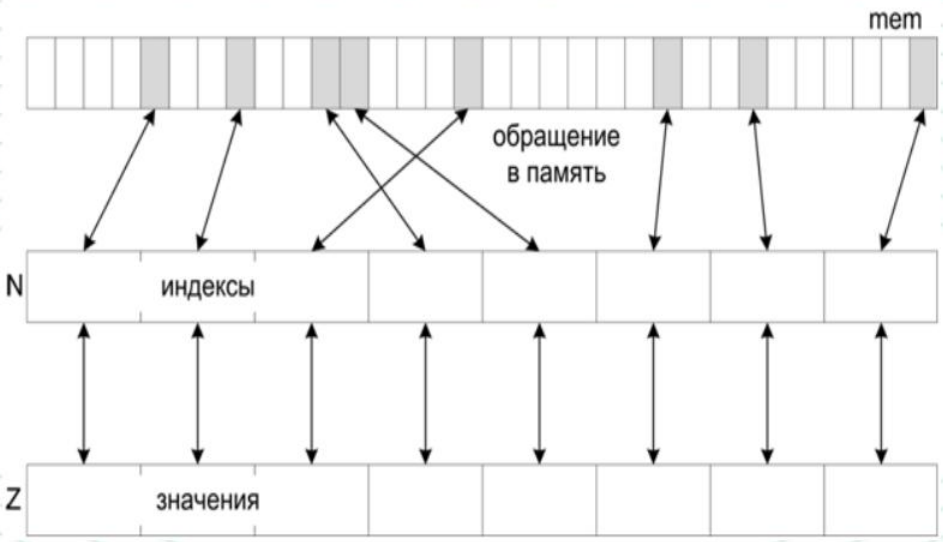
_mm512_mask_i32scatter_ps

_mm512_mask_sub_epi32

_mm512_mask_cmp_epi32_mask

_mm512_mask_load_ps

Интринсики, используемые в коде векторизации



_mm512_mask_i32scatter_ps

Формулы для расчета числа итераций

$$T = \sum_{k \in ks} \sum_{i=k}^{n-1} I(k,i)$$

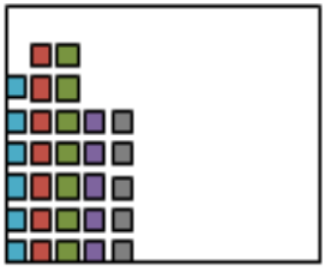
$$T_v = \sum_{k \in ks} \left(\left(\sum_{g=0}^{G(k)-1} \max_{i=k+w(k)g}^{k+w(k)(g+1)-1} I(k,i) \right) + \max_{i=k+w(k)G(k)}^{n-1} I(k,i) \right)$$

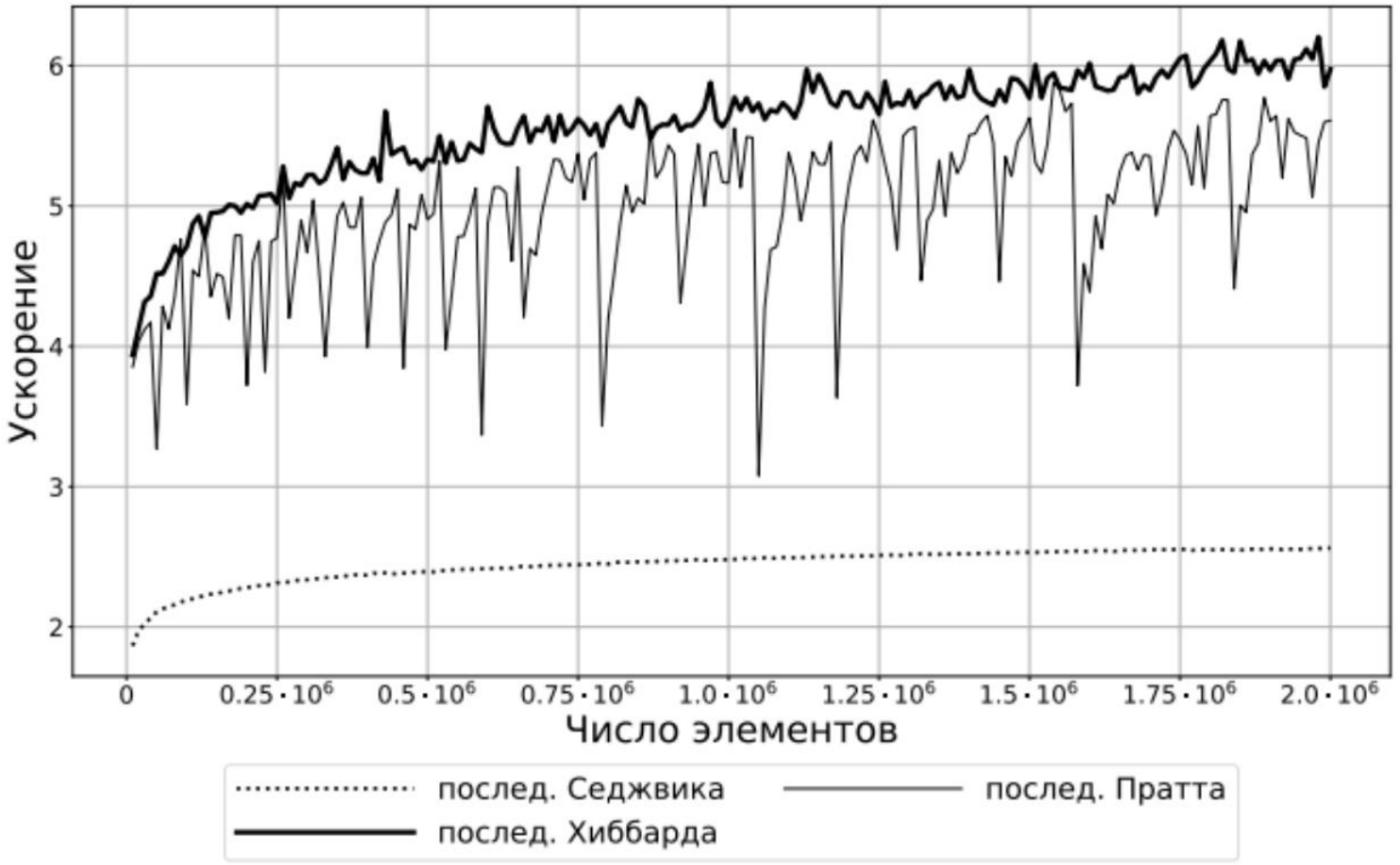
где $w(k) = \min(k, 16)$, $G(k) = \lfloor \frac{n-k}{w(k)} \rfloor$

Исходный массив

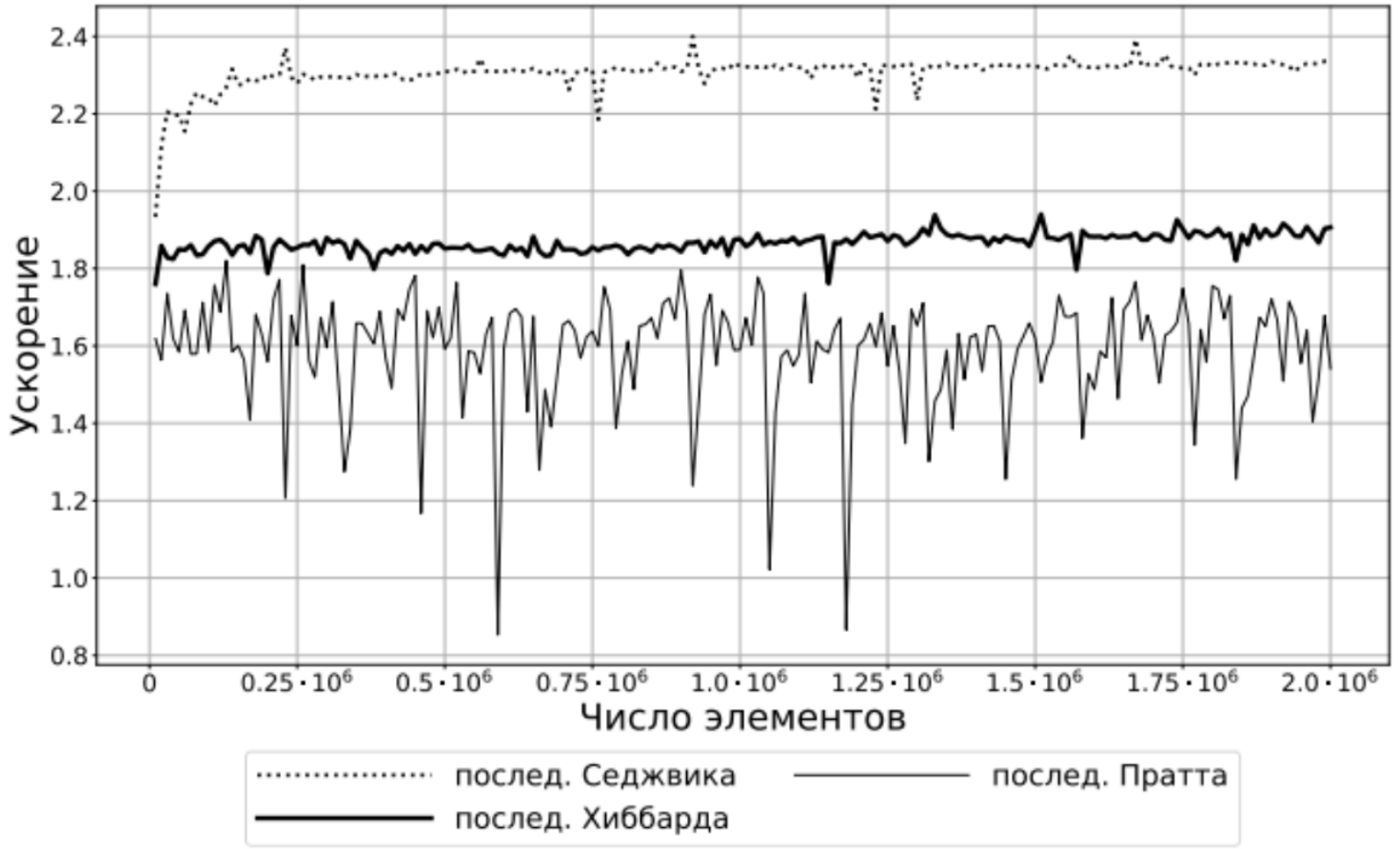
I

Векторизация

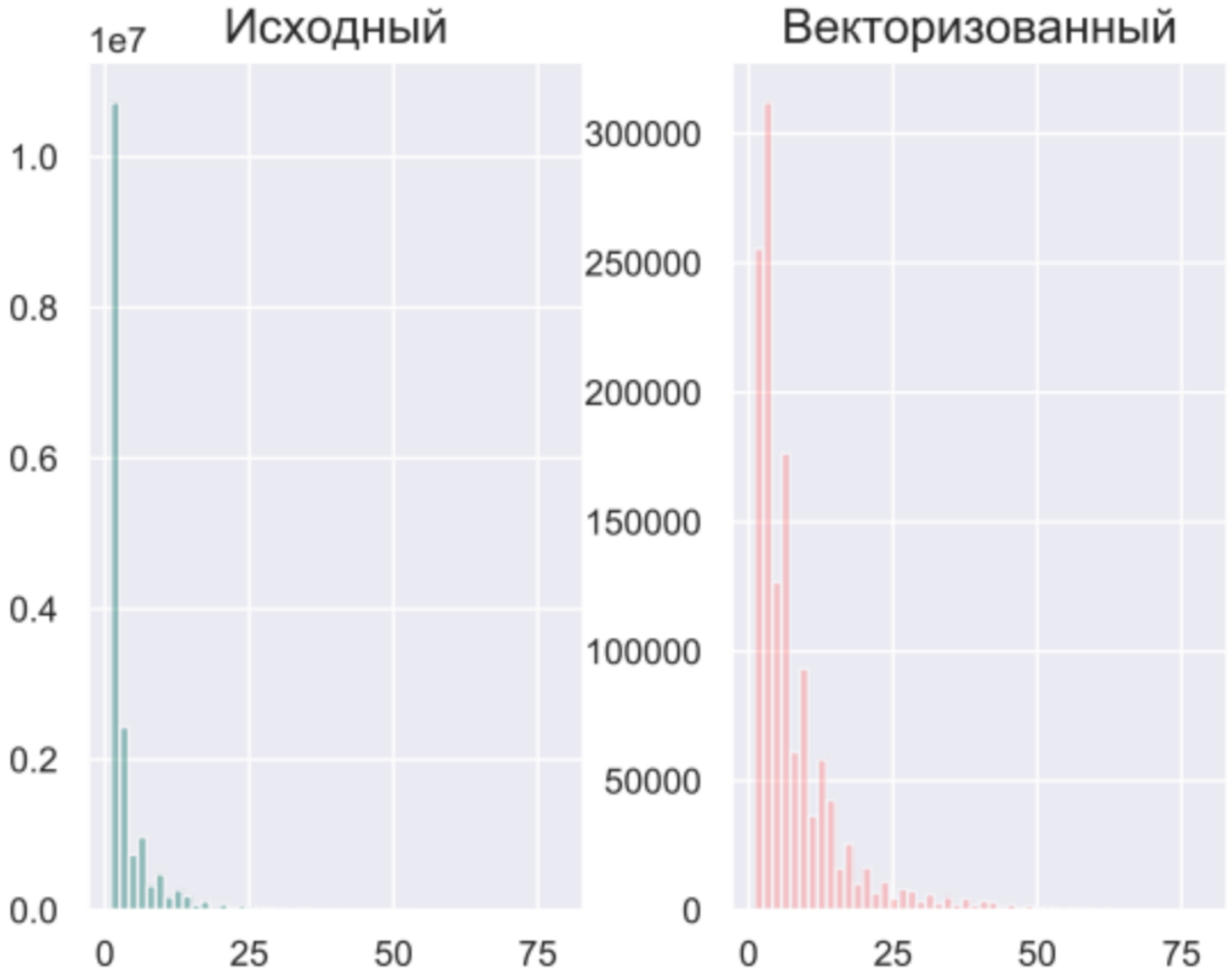




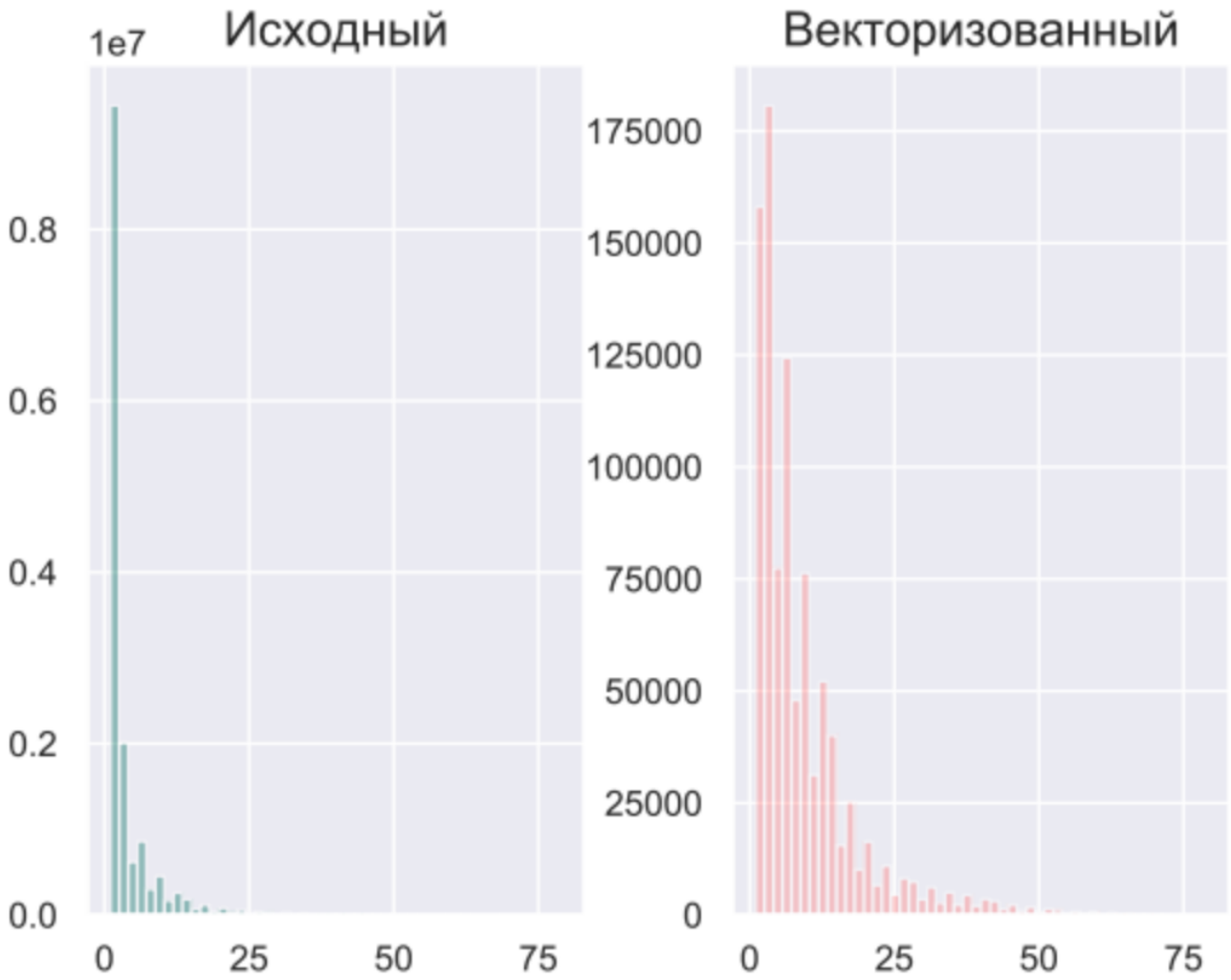
Сравнение **теоретического ускорения** векторизованной версии сортировки Шелла для различных последовательностей шагов



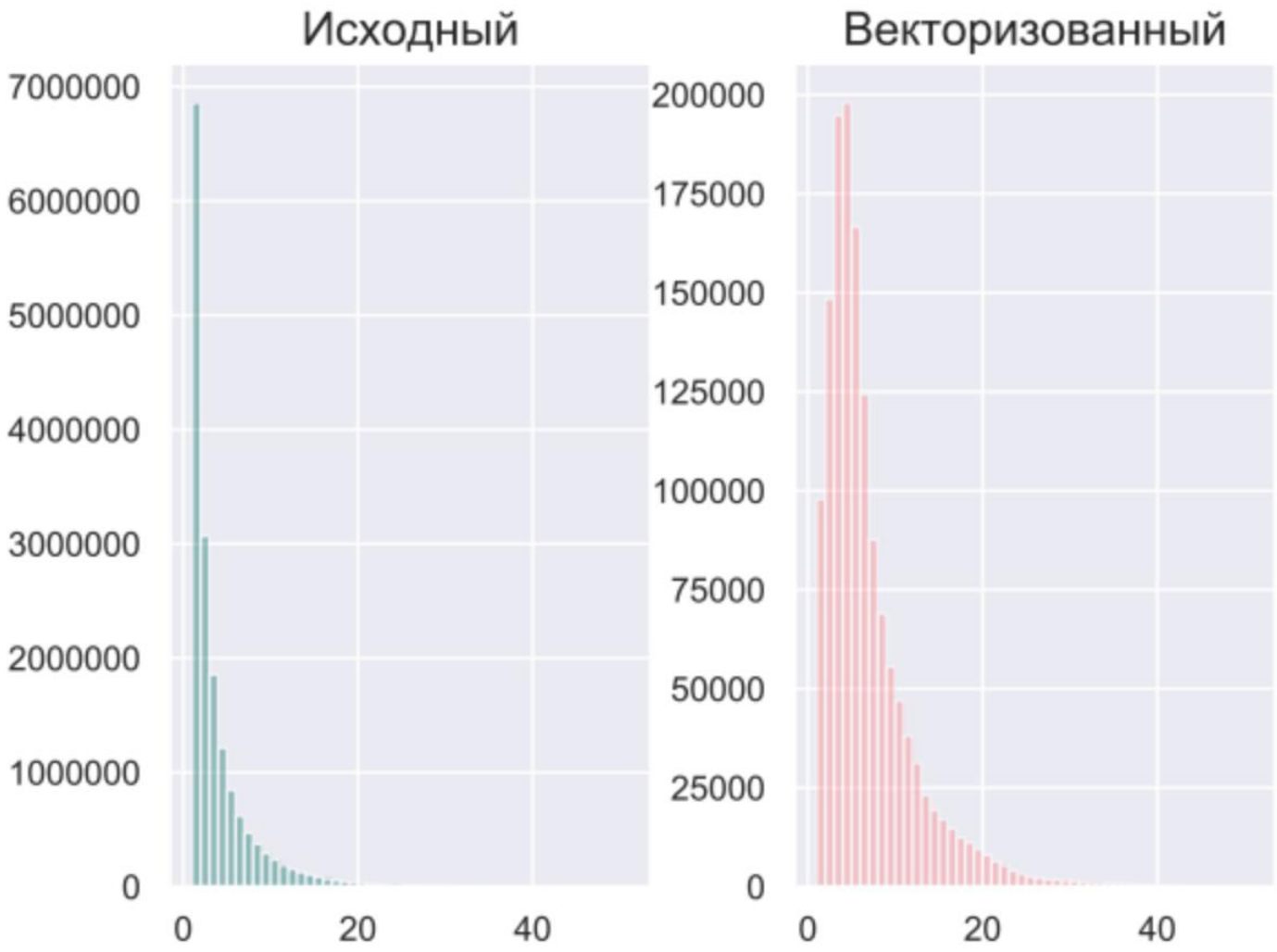
Сравнение **экспериментального ускорения** векторизованной версии сортировки Шелла для различных последовательностей шагов



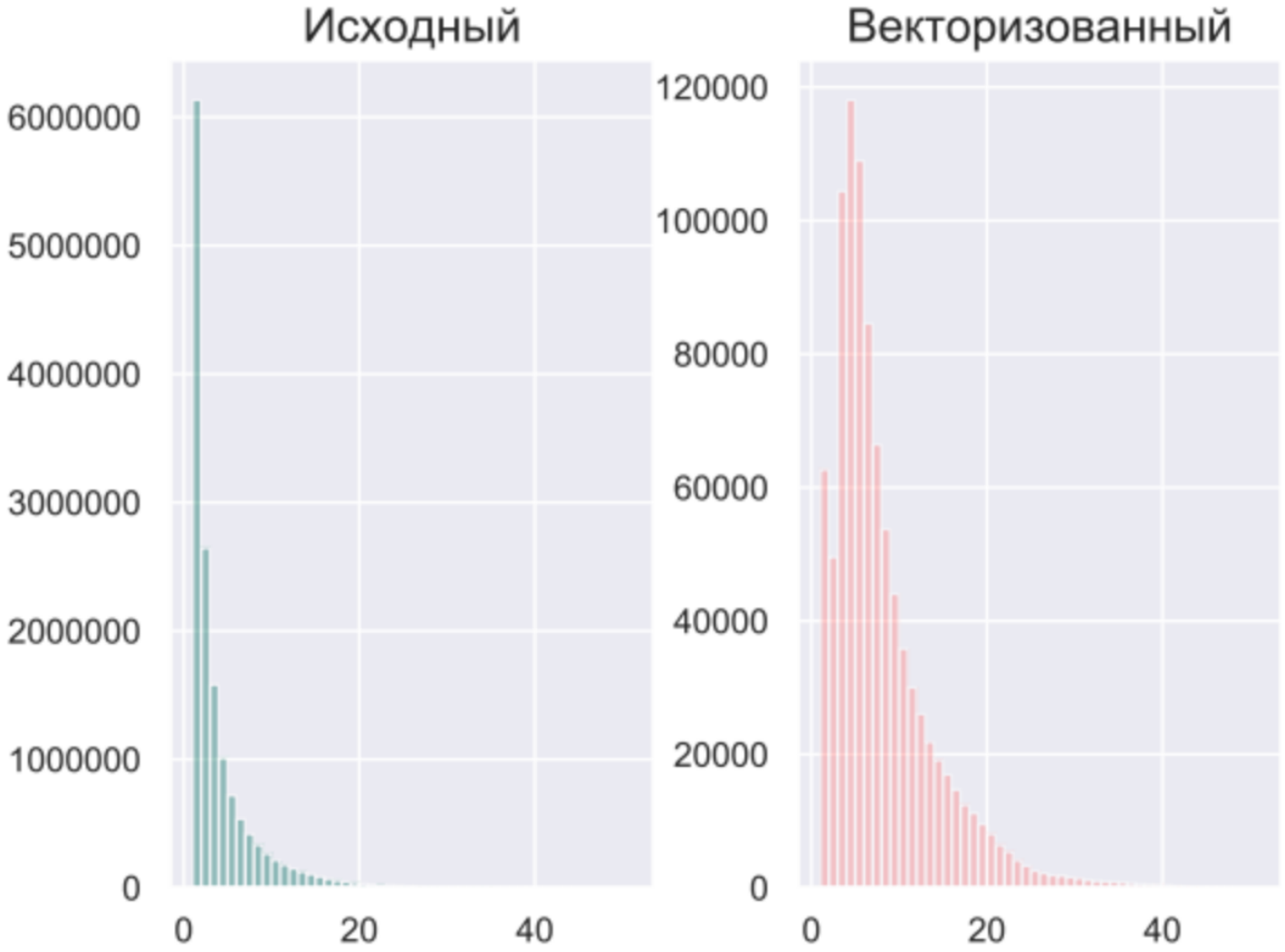
Гистограмма распределения **количества итераций** внутреннего цикла при сортировке с последовательностью **Шелла при k = 4**



Гистограмма распределения **количества итераций** внутреннего цикла при сортировке с последовательностью **Шелла при k = 15**



Гистограмма распределения **количества итераций** внутреннего цикла при сортировке с последовательностью Хиббарда **при k = 3**



Гистограмма распределения **количества итераций** внутреннего цикла при сортировке с последовательностью Хиббарда **при k = 15**

Выводы

- Для гнезд циклов с нерегулярным числом итераций теоретическое и экспериментальное ускорение далеко от идеальной верхней границы
- Эффективность векторизации стремиться к асимптотическому значению
- Необходимость оценки характера исполнения дискретных задач в каждом отдельном случае

СПАСИБО ЗА ВНИМАНИЕ

shumilin@jscc.ru
noisd@yandex.ru
+7(905)502-73-66