High-throughput drug discovery on the Fujitsu A64FX

Filippo **Barbari**, Federico **Ficarelli**, Daniele **Cesarini**

Cineca

IWAHPCE-2024 @ HPC-Asia 2024

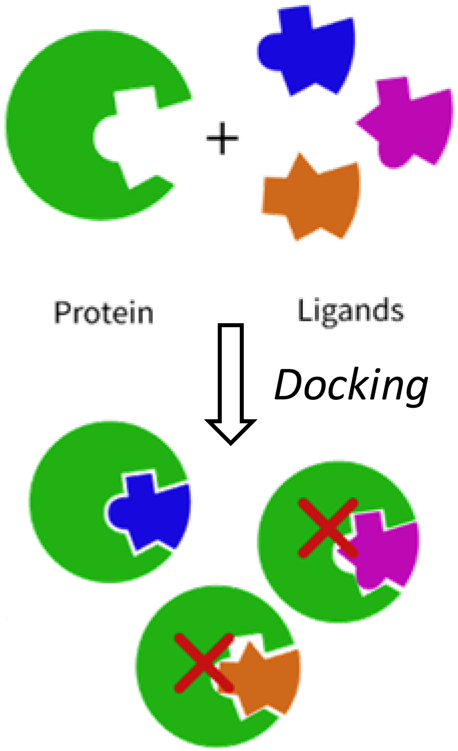
25/1/2024, Nagoya, Japan

EUPEX

European Pilot for Exascale

Drug Discovery via Virtual Screening

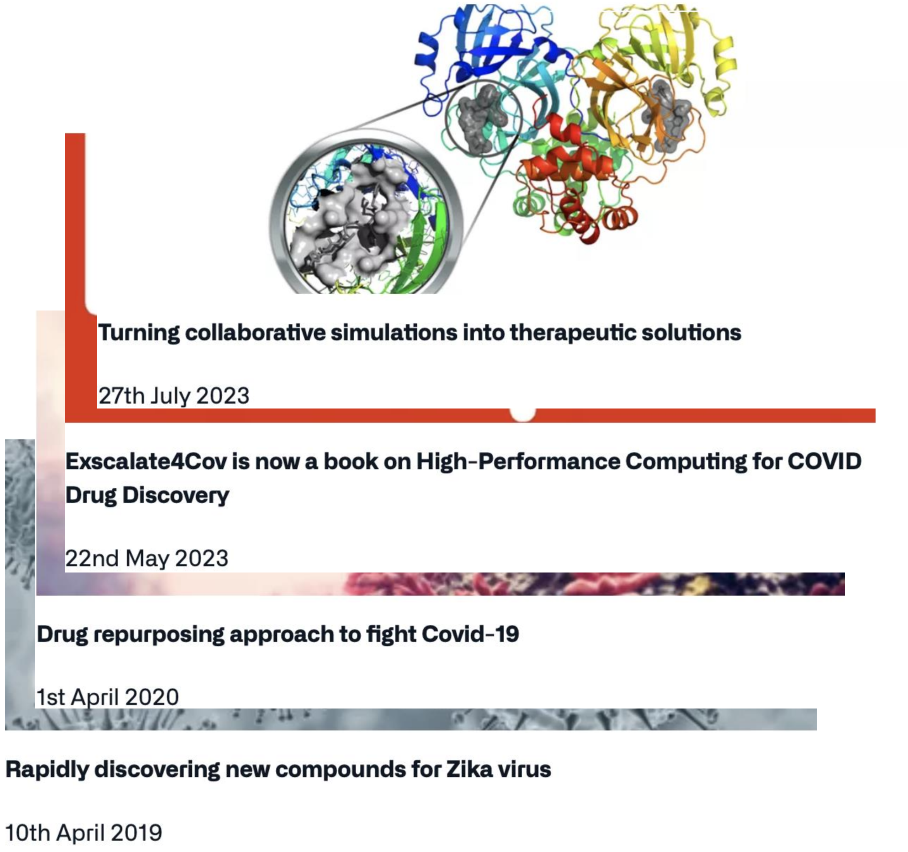
- Discovering and bringing a new drug to market can take **10 years** and cost **$1B**.
- **CADD (*Computer Aided Drug Design*)** accelerates such process by selecting only those molecules likely to be effective against the disease or condition.
- Virtual screening (or **docking**) attempts to bind the proposed drug molecule (*ligand*) into spaces present in the surface of a protein associated with the disease (*target*) .
- Goal: given a candidate compound (ligand), estimate the likelihood of interaction w/the target protein



Ligand docked in protein (a complex)

Virtual screening: HPC to the aid

- **Key idea**: chemical space exploration as a game changer for the researcher
- Dominated by sheer throughput
- **Issue**: commercial/OSS tools focus on feature completeness, unsuitable for extreme HPC scale-out
- Cineca and Dompé Pharmaceuticals teamed up in late ’90 for a custom solution[1]
- Models and features focused only on drug discovery
- Made large scale campaigns against Zika (2018) and COVID-19 (2020) feasible



Turning collaborative simulations into therapeutic solutions

27th July 2023

Exscalate4Cov is now a book on High-Performance Computing for COVID Drug Discovery

22nd May 2023

Drug repurposing approach to fight Covid-19

1st April 2020

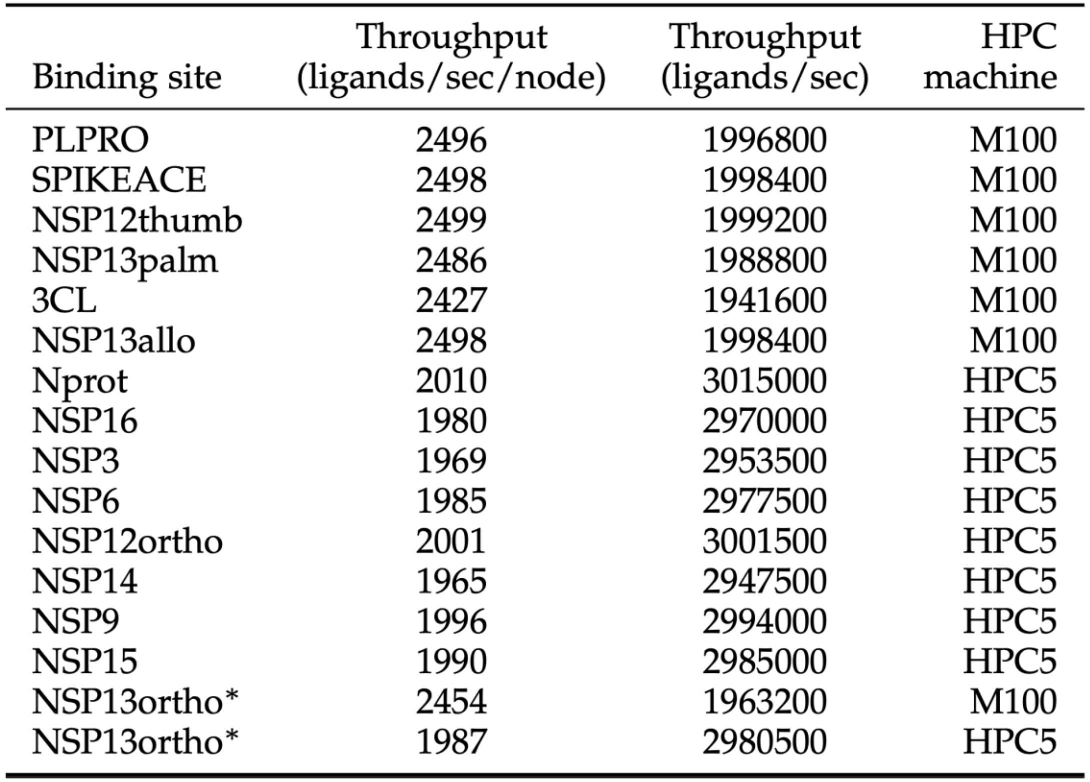
Rapidly discovering new compounds for Zika virus

10th April 2019

[1] Beccari, A. R.; Cavazzoni, C.; Beato, C.; Costantino, G. **LiGen: A High Performance Workflow for Chemistry Driven de Novo Design**. J. Chem. Inf. Model. 2013, 53 (6), 1518–1527. https://doi.org/10.1021/ci400078g.

Exscalate4CoV: HPC against COVID-19

- Virtual screened more than **70 B ligands** against **16 binding sites** of 12 proteins of SARS-Cov-2
- 60 hours to perform **a trillion docking ops**
- Learned a lot about the **hurdles** of ***urgent computing***
- **Largest virtual screening experiment to date** with 50x more ligands and 7.5x more targets than the previous record[1,2]
- Safe drug (Raloxifene) found to be active when repurposed against mild COVID-19 symptoms, reached clinical trials[3]



Binding site	Throughput (ligands/sec/node)	Throughput (ligands/sec)	HPC machine
PLPRO	2496	1996800	M100
SPIKEACE	2498	1998400	M100
NSP12thumb	2499	1999200	M100
NSP13palm	2486	1988800	M100
3CL	2427	1941600	M100
NSP13allo	2498	1998400	M100
Nprot	2010	3015000	HPC5
NSP16	1980	2970000	HPC5
NSP3	1969	2953500	HPC5
NSP6	1985	2977500	HPC5
NSP12ortho	2001	3001500	HPC5
NSP14	1965	2947500	HPC5
NSP9	1996	2994000	HPC5
NSP15	1990	2985000	HPC5
NSP13ortho*	2454	1963200	M100
NSP13ortho*	1987	2980500	HPC5

[1] Gadioli, D., Vitali, E., Ficarelli, F., Latini, C., Manelfi, C., Talarico, C., Silvano, C., Cavazzoni, C., Palermo, G., Beccari, A.R., 2022**. EXSCALATE: An Extreme-Scale Virtual Screening Platform for Drug Discovery Targeting Polypharmacology to Fight SARS-CoV-2**. IEEE Trans. Emerg. Topics Comput. 1–12. https://doi.org/10.1109/TETC.2022.3187134

[2] S. LeGrand et al., **GPU-Accelerated Drug Discovery with Docking on the Summit Supercomputer: Porting, Optimization, and Application to COVID-19 Research**, in Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics, Virtual Event USA: ACM, Sep. 2020, pp. 1–10. doi: 10.1145/3388440.3412472.

[3] E. Nicastri et al., **A phase 2 randomized, double-blinded, placebo-controlled, multicenter trial evaluating the efficacy and safety of raloxifene for patients with mild to moderate COVID-19**, eClinicalMedicine, vol. 48, p. 101450, Jun. 2022, doi: 10.1016/j.eclinm.2022.101450

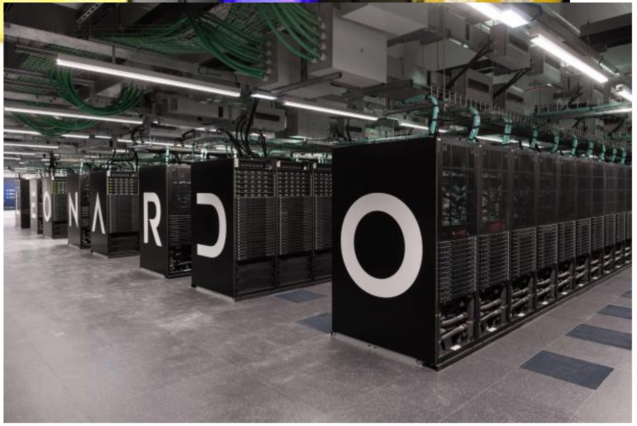
The A64FX as a production target

- GPU kernels optimized and tuned on NVIDIA[1]
- CPU kernels must be as efficient as possible on all production targets
- Current: AVX512, PowerPC VSX
- Fujitsu A64FX elected as the SVE software development vehicle for RHEA European Processor Initiative by EUPEX European Pilot for Exascale

Marconi100 @ Cineca

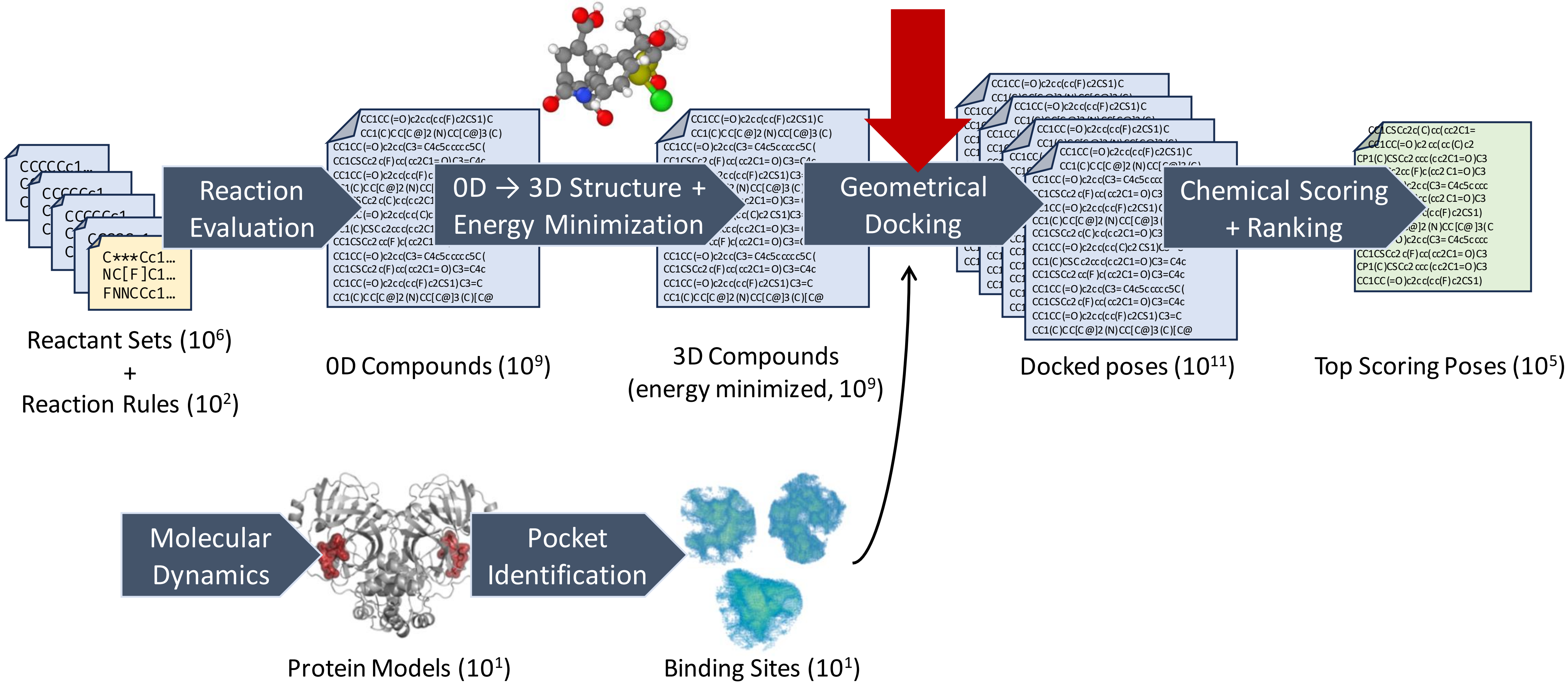
HPC5 @ ENI

RHEA Pilot @ EUPEX


Leonardo @ Cineca

[1] Vitali, E.; Ficarelli, F.; Bisson, M.; Gadioli, D.; Accordi, G.; Fatica, M.; Beccari, A. R.; Palermo, G. **GPU-Optimized Approaches to Molecular Docking-Based Virtual Screening in Drug Discovery: A Comparative Analysis**. Journal of Parallel and Distributed Computing 2023, 104819. https://doi.org/10.1016/j.jpdc.2023.104819

High-throughput pipeline



Reaction Evaluation

0D → 3D Structure + Energy Minimization

Geometrical Docking

Chemical Scoring + Ranking

Molecular Dynamics

Pocket Identification

Reactant Sets (10^6)
+
Reaction Rules (10^2)

0D Compounds (10^9)

3D Compounds
(energy minimized, 10^9)

Docked poses (10^{11})

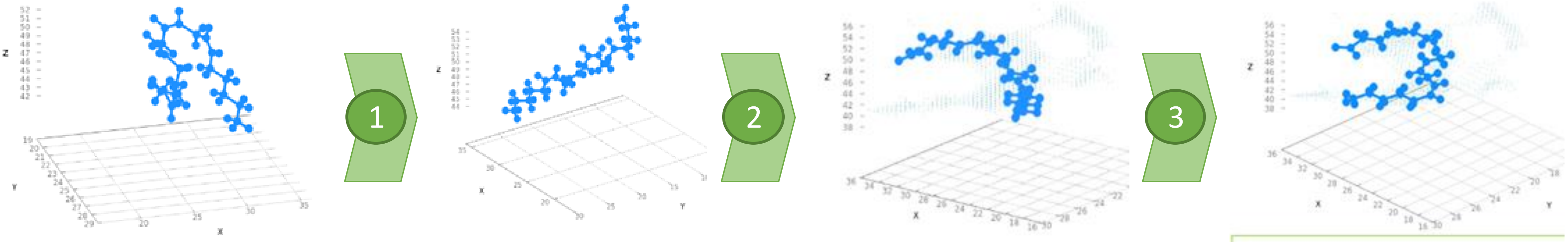
Top Scoring Poses (10^5)

Protein Models (10^1)

Binding Sites (10^1)

Docking: qualitative stage

Semi-flexible geometric docking using the Fischer's **lock-and-key model:**



1. Ligand Expansion

- Identification of the rotatable bonds
- Internal distances maximization

2. Initial Placement

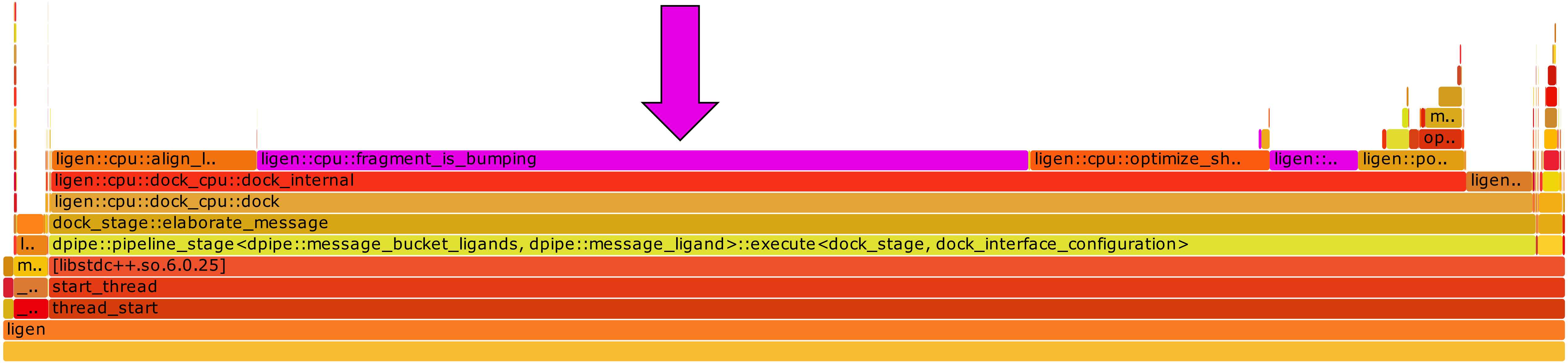
- Ligand main fragments decomposition
- Ligand initial poses identification
- Placement of the ligand into the pocket with rigid roto-translations

3. Shape Refinement

- Use of the rotatable bonds to modify the ligand shape and to match the protein pocket
- Maximizing docking score

The problem

49,38% of the total execution time is spent in a single function, `fragment_is_bumping`. Let's take a look.



The problem

```
Input: coords, mask, length
for i = 0; i < length; i = i + 1 do
    for j = i + 1; j < length; j = j + 1 do
        if mask[i] AND mask[j] then
            d = distance(coords[i], coords[j]);
            if d < limitDistance then
                return True;
            end
        end
    end
end
return False;
```

Only scalar 64-bit registers are being used!

Clang++ 16.0.6
-O3 -DNDEBUG -mcpu=a64fx

```
.outer_loop_body:
    // ...outer loop control flow...
.inner_loop_header:
    add     x4, x4, #8
    add     x2, x2, #1
    subs    x3, x3, #1
    b.eq    .outer_loop_body
.inner_loop_body:
    ldrb    w5, [x2]
    tst     w5, w16
    b.eq    .inner_loop_body // mask is zero
    ldr     d1, [x0, x12, lsl #3]
    ldr     d4, [x3, #8]
    fsub    d1, d1, d4
    ldr     d2, [x17]
    ldr     d5, [x3, #1544]
    fsub    d2, d2, d5
    ldr     d3, [x18]
    ldr     d6, [x3, #3080]
    fsub    d3, d3, d6
    fmul    d1, d1, d1
    fmadd   d1, d2, d2, d1
    fmadd   d1, d3, d3, d1
    fcmp    d1, d0
    b.ge    .inner_loop_body
    mov     w14, w13
    and     w0, w14, #0x1
    ret
```

Highway library

An abstraction layer for **cross-platform SIMD intrinsics**.

Dual-licensed: Apache-2.0 + BSD-3

Supports up to 20 targets:

- ARM: NEON, SVE
- Risc-V: RVV 1.0
- IBM: PPC8, PPC9
- X86: SSE2, SSSE3, SSE4, AVX2, AVX512

Supports static dispatch (architecture detection through compiler's −`m` flags) and dynamic dispatch (multiple targets in compilation, target selection at runtime).

Currently used in the JPEG XL codec reference implementation, NumPy and TensorFlow.

https://github.com/google/highway

An example

```
double ddot(const double* __restrict__
a,                      const double*
__restrict__ b,
            const size_t len) {
    double s{0.0};
    for (size_t i{0u}; i < len; i++) {
        s += a[i] * b[i];
    }
    return s;
}
```

```
HWY_ATTR double ddot(const double* HWY_RESTRICT a,
                     const double* HWY_RESTRICT b,
                     const size_t len) {
    namespace hn = hwy::HWY_NAMESPACE;
    const hn::ScalableTag<double> d;
    using V = hn::Vec<decltype(d)>;
    const size_t lanes = hn::Lanes(d);

    V s = hn::Zero(d);
    size_t i{0u};
    for (; i + (lanes - 1) < len; i += lanes) {
        const V ai = hn::LoadU(d, a + i);
        const V bi = hn::LoadU(d, b + i);
        s = hn::MulAdd(ai, bi, s);
    }
    if (i < lanes) {
        const V ai = hn::LoadN(d, a + i, lanes - i);
        const V bi = hn::LoadN(d, b + i, lanes - i);
        s = hn::MulAdd(ai, bi, s);
    }
    return hn::GetLane(hn::SumOfLanes(d, s));
}
```

The fragment_is_bumping micro-kernel

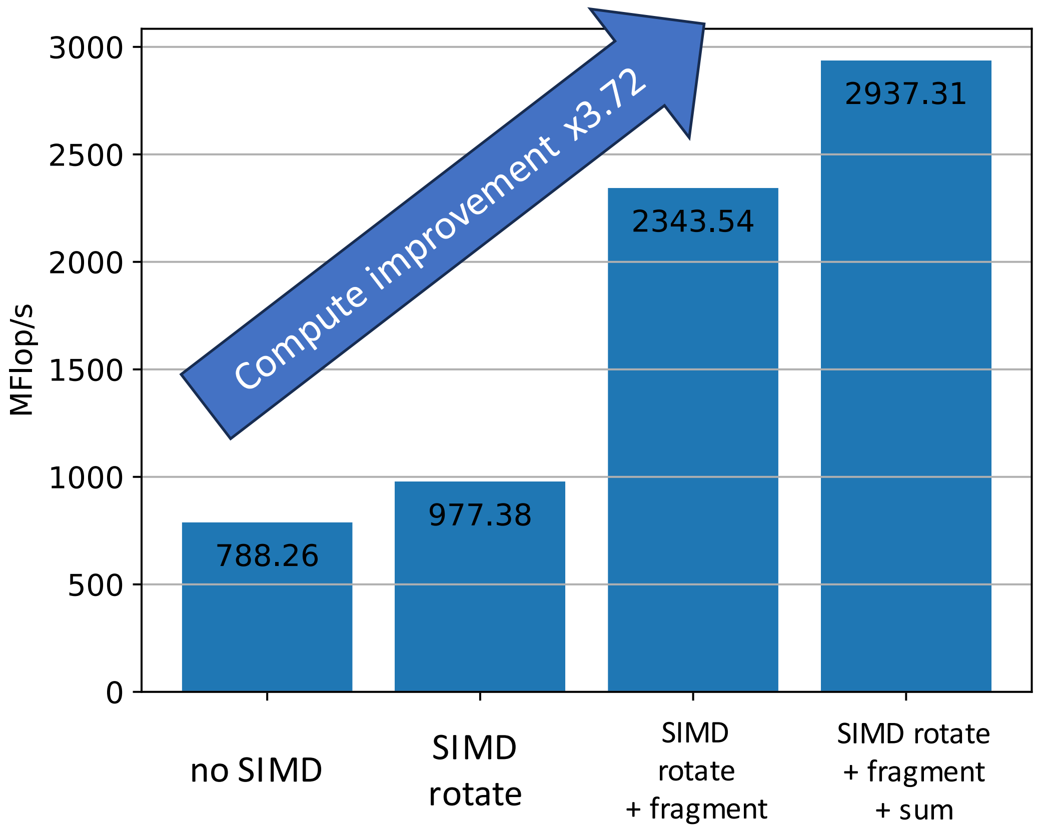
`clang++-16.0.6 -O3 -DNDEBUG -mcpu=a64fx`

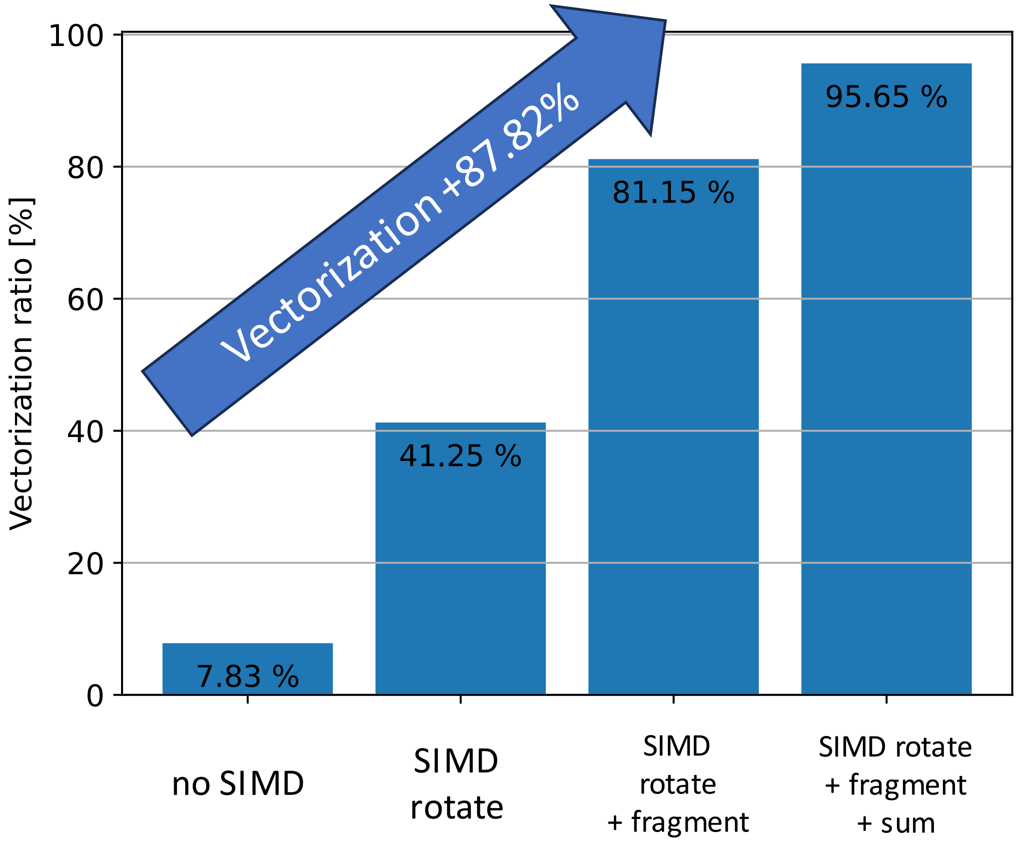
```
.outer_loop_body:
    // ...outer loop control flow...
.inner_loop_header:
    add     x4, x4, #8
    add     x2, x2, #1
    subs    x3, x3, #1
    b.eq    .outer_loop_body
.inner_loop_body:
    ldrb    w5, [x2]
    tst     w5, w16
    b.eq    .inner_loop_body // mask is zero
    ldr     d1, [x0, x12, lsl #3]
    ldr     d4, [x3, #8]
    fsub    d1, d1, d4
    ldr     d2, [x17]
    ldr     d5, [x3, #1544]
    fsub    d2, d2, d5
    ldr     d3, [x18]
    ldr     d6, [x3, #3080]
    fsub    d3, d3, d6
    fmul    d1, d1, d1
    fmadd   d1, d2, d2, d1
    fmadd   d1, d3, d3, d1
    fcmp    d1, d0
    b.ge    .inner_loop_body // bumping check clear
    mov     w14, w13
    and     w0, w14, #0x1
    ret
```

`clang++-16.0.6 -O3 -DNDEBUG -mcpu=a64fx`

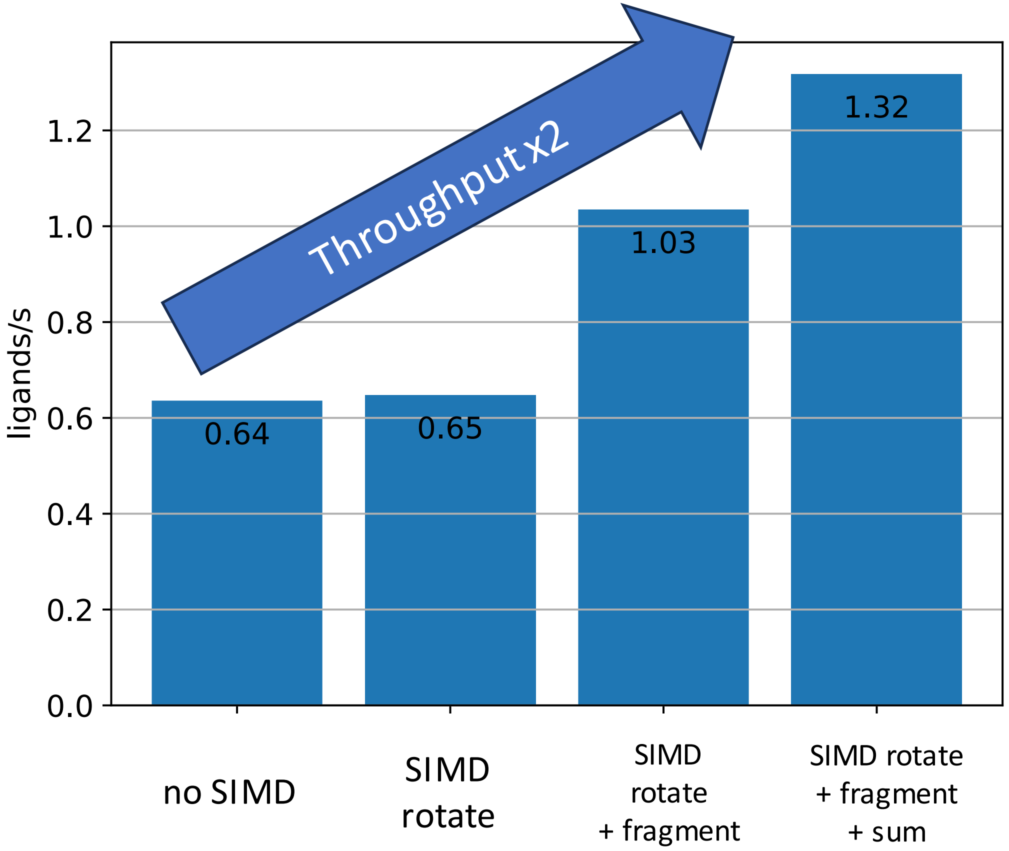
```
...
    whilelo p2.d, wzr, w7
    cmp     x9, x6
    csel    x7, x9, x6, lo
    cmp     x7, #32
    csel    x7, x7, x14, lo
    whilelo p3.b, wzr, w7
    add     x7, x0, x13, lsl #3
    ld1d    { z16.d }, p2/z, [x7, x17, lsl #3]
    ld1b    { z6.b }, p3/z, [x10, x13]
    fsub    z16.d, z4.d, z16.d
    and     z7.b, p1/m, z7.b, z6.b
    uunpklo z6.h, z7.b
    ld1d    { z7.d }, p2/z, [x7, x16, lsl #3]
    fmul    z16.d, z16.d, z16.d
    uunpklo z6.s, z6.h
    fsub    z7.d, z3.d, z7.d
    uunpklo z6.d, z6.s
    cmpne   p3.d, p1/z, z0.d, z6.d
    ld1d    { z6.d }, p2/z, [x7, x15, lsl #3]
    fmad    z7.d, p1/m, z7.d, z16.d
    fsub    z6.d, z2.d, z6.d
    fmad    z6.d, p1/m, z6.d, z7.d
    fcmgt   p4.d, p1/z, z1.d, z6.d
    and     p3.b, p4/z, p4.b, p3.b
    and     p2.b, p3/z, p3.b, p2.b
    ptest   p0, p2.b
...
```

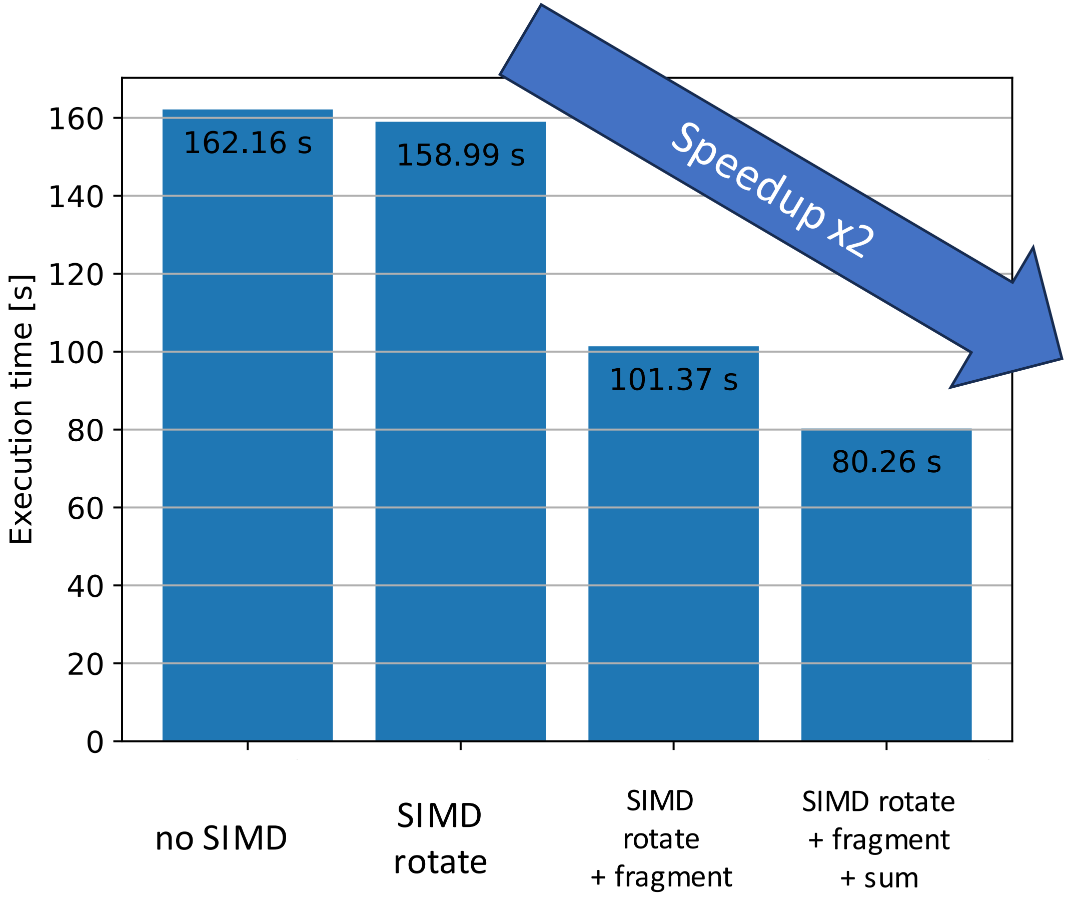
The results – FLOPS & Vectorization



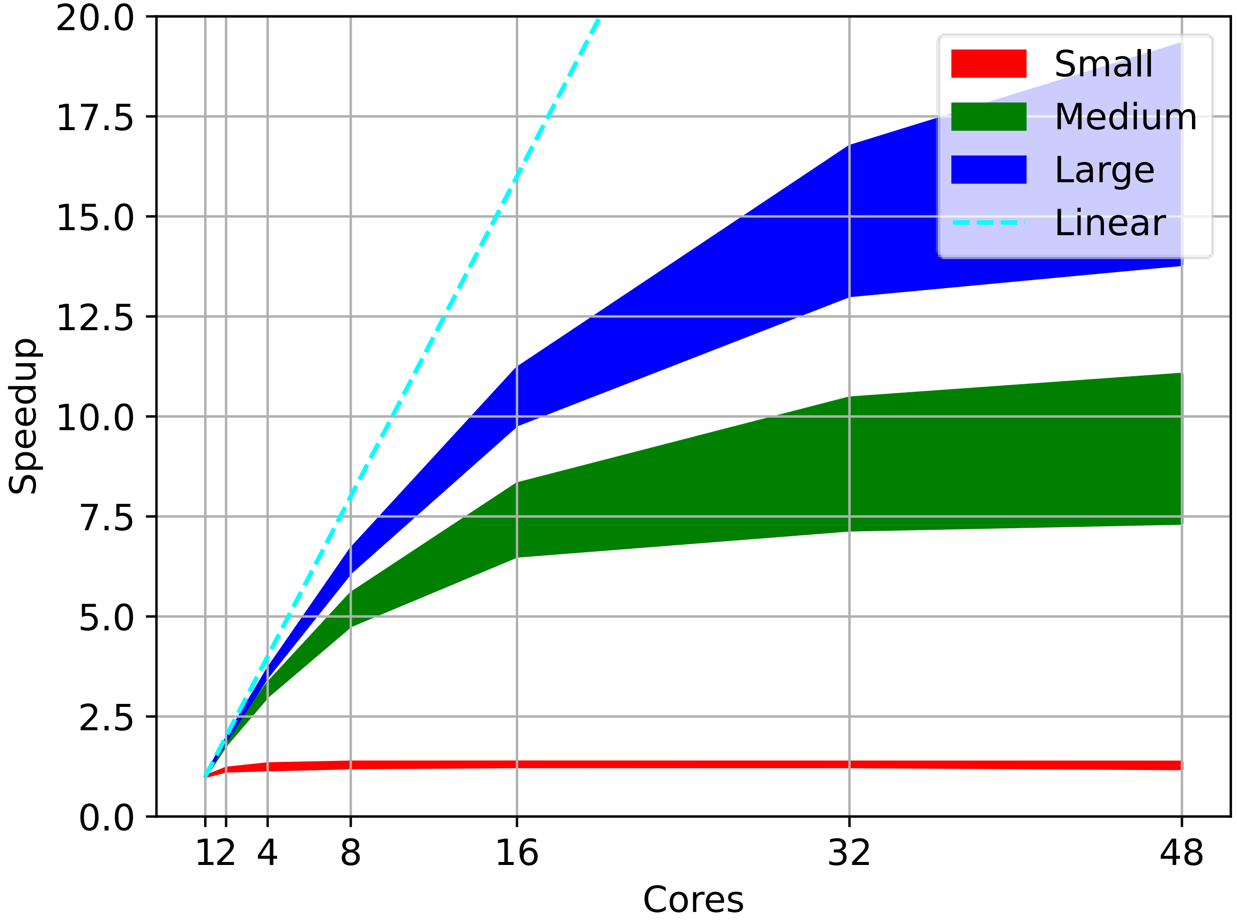


The results – Throughput & Speedup

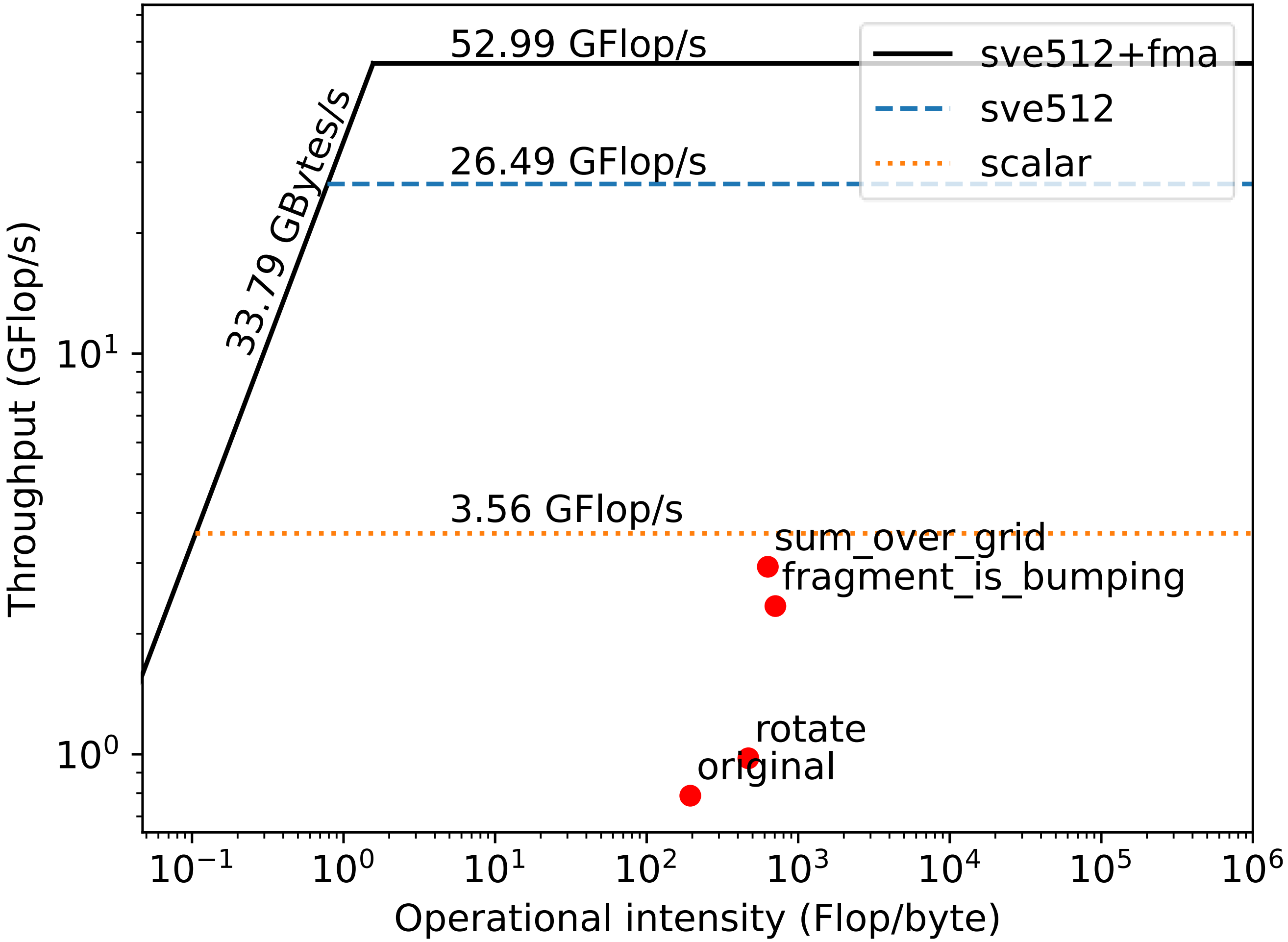




The results – CPU speedup



Work in progress



Thanks for your attention

Filippo **Barbari**, Federico **Ficarelli**, Daniele **Cesarini**

Cineca

IWAHPCE-2024 @ HPC-Asia 2024

25/1/2024, Nagoya, Japan