Optimization of a Stencil Computation considering the Architecture of SX-Aurora TSUBASA

31st Workshop on Sustained Simulation Performance (WSSP31)

Kazuhiko Komatsu

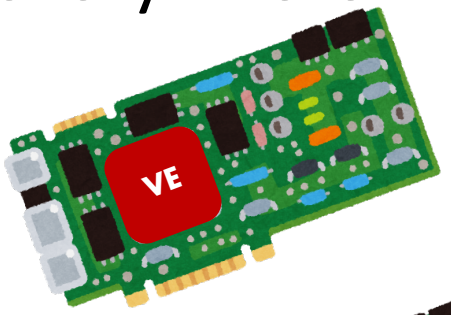
2021/3/18

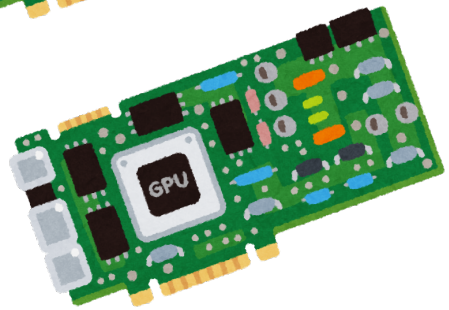
Agenda

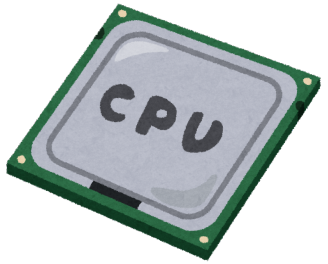
- Background
- Objective and approach
- Optimizations of a stencil computation
- Evaluation
- Conclusions and future work

Introduction

- Vector computing capability everywhere
  - Vector processor
    - NEC SX series
      - 256 words (2KB =16 Kbits) vector processing
  - GPU
    - Massive parallel computing in a kind of a vector computing way
    - NVIDIA GPU, AMD GPU
      - SIMT
  - CPU
    - Vector computing capability support
    - Intel Xeon, Fujitsu A64FX
      - 512 bits vector processing

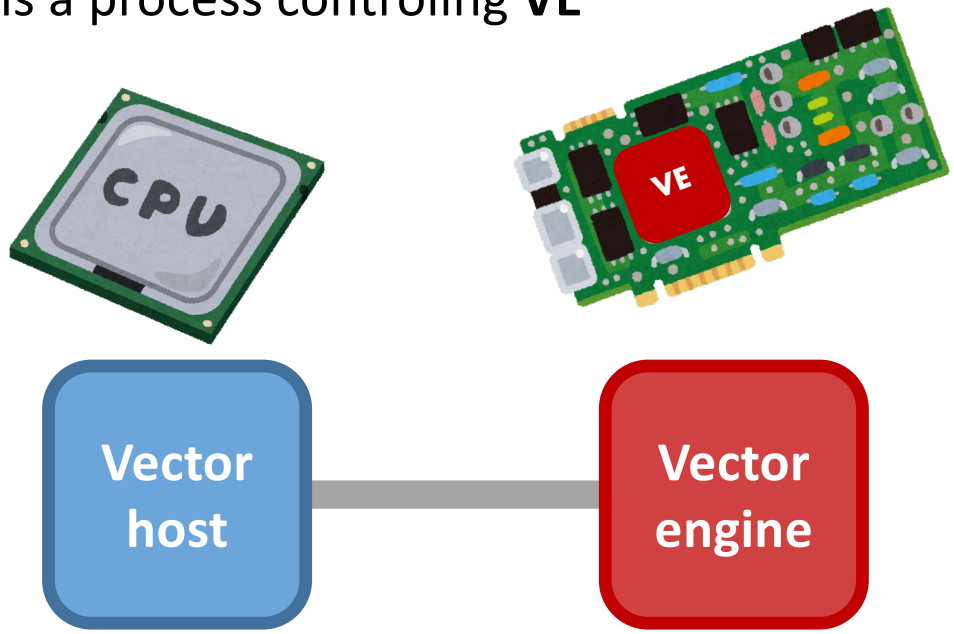




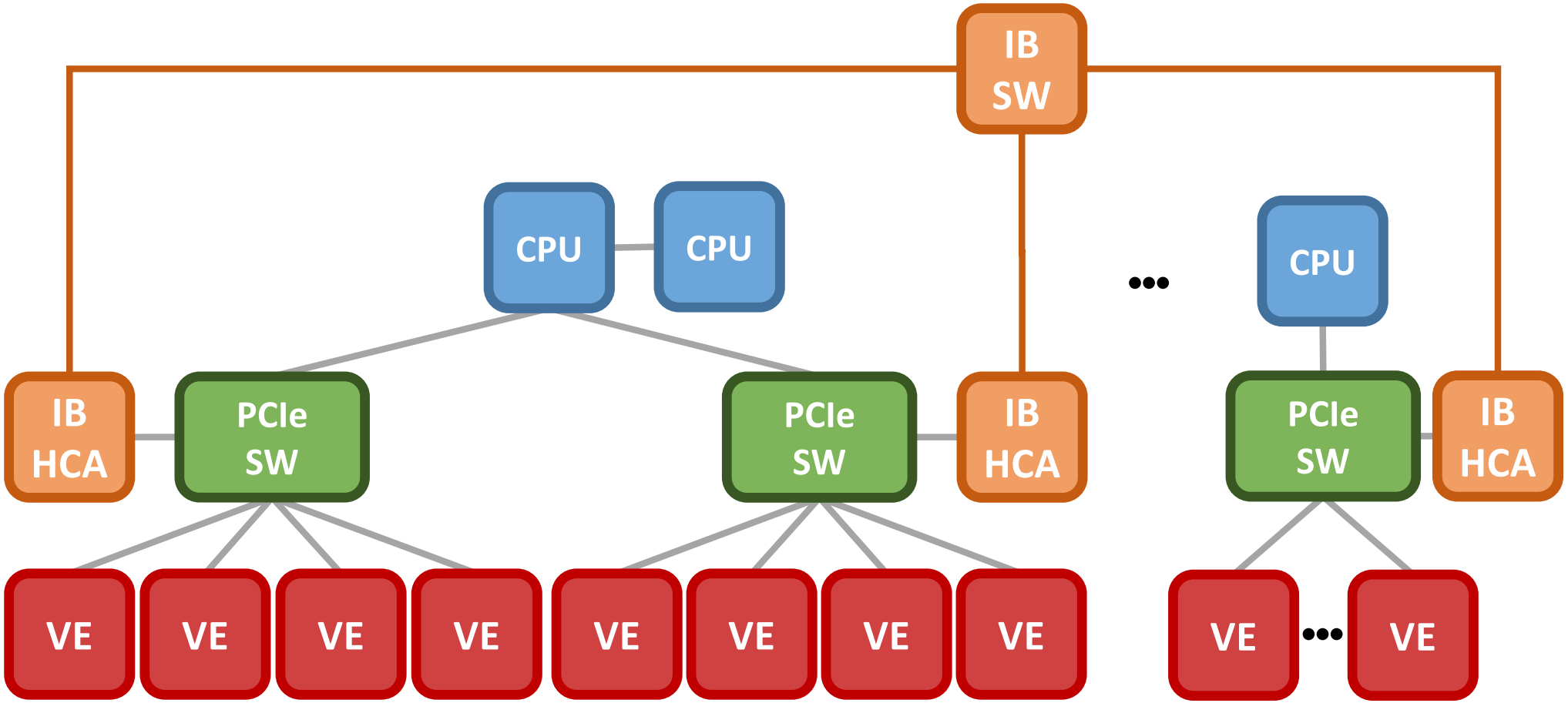


Modern vector computer

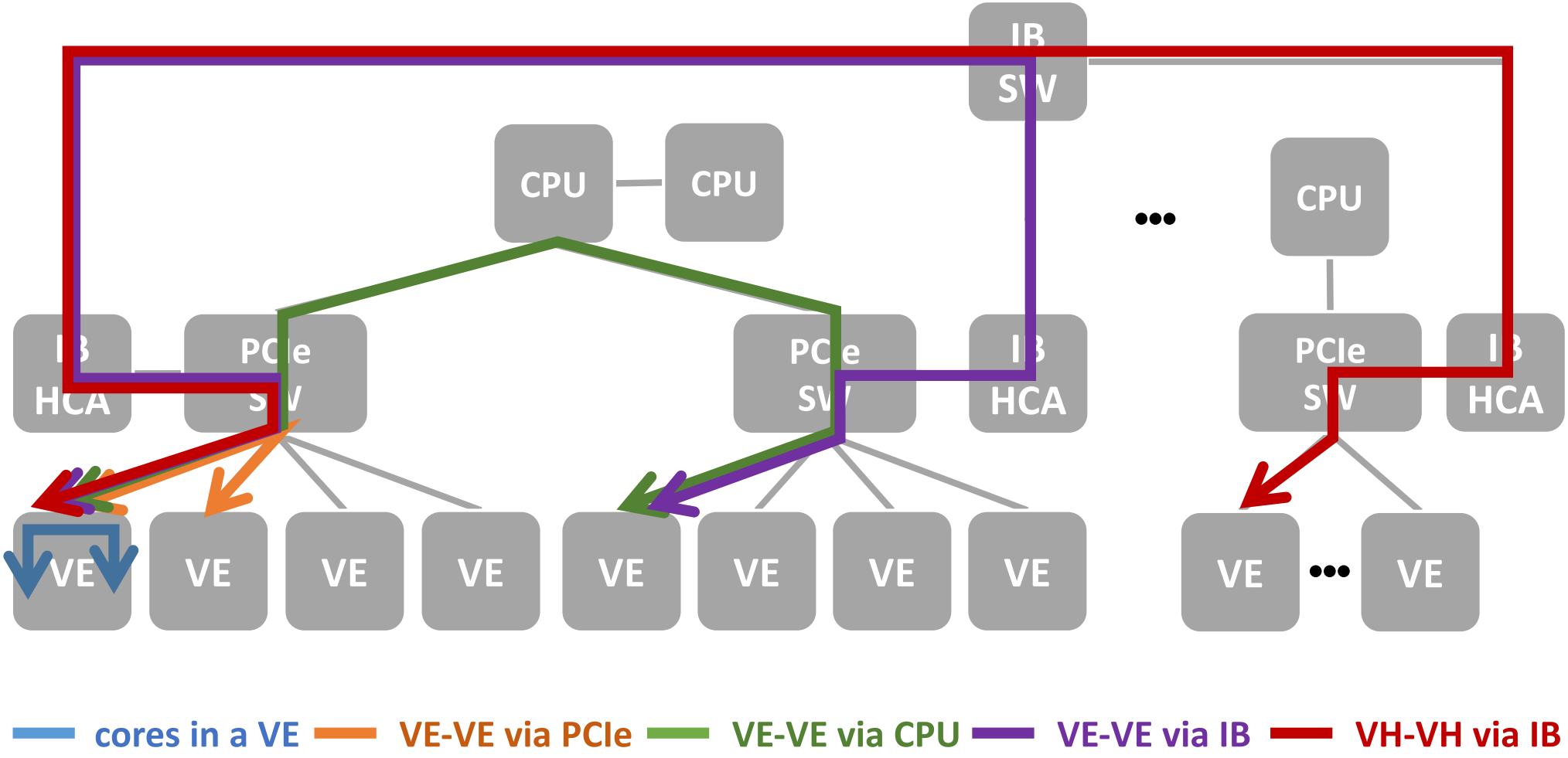
- SX-Aurora TSUBASA
  - **Vector engine (VEs)** are attached to a **vector host (VH)**
    - **VE** is responsible for the execution of an application
    - **VH** runs a process controling **VE**



SX-Aurora TSUBASA (A300-8)



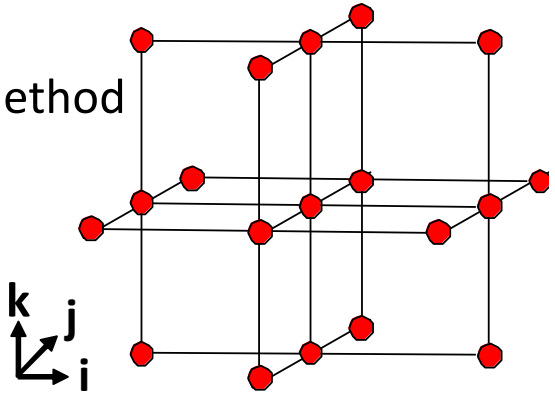
Communication patterns in A300-8



Need to consider the hierarchy to expolit full performance of a system

Objective

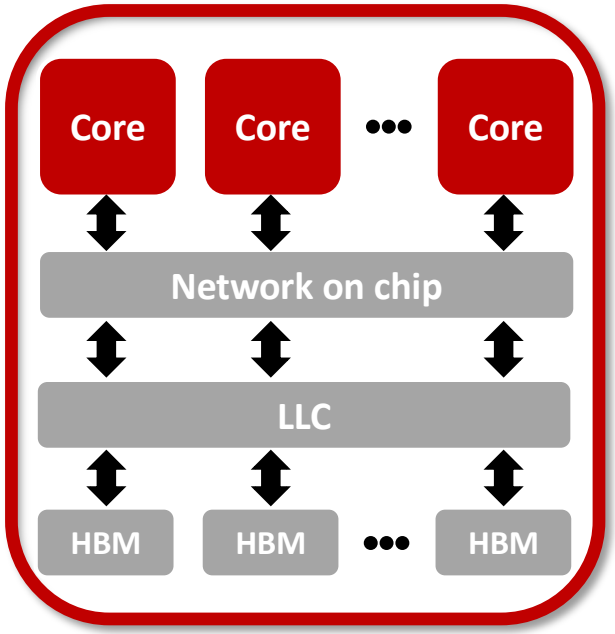
- Objective
  - Exploit high performance of such a computing system
    - Himeno benchmark as a stencil computation
      - An incompressible fluid analysis code
      - Poisson's equation using the Jacobi iteration method
      - 19-point stencil calculation
    - Memory-bound computation

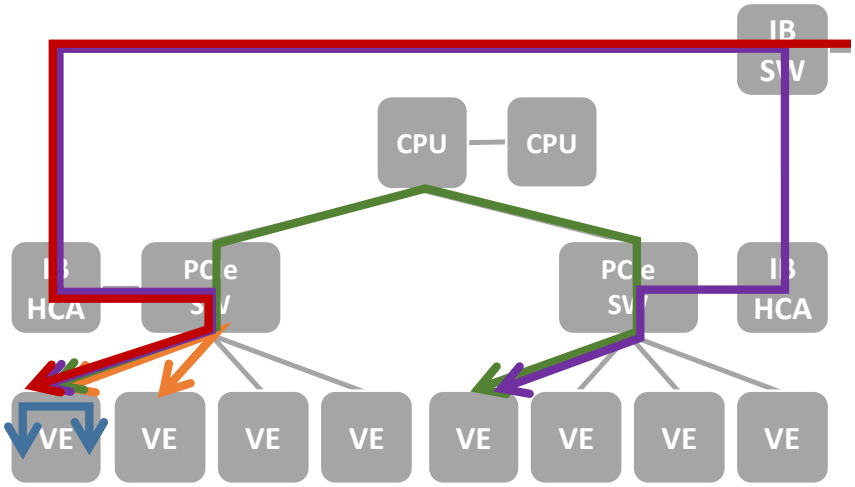


- Approach
  - Exploit various bandwidths of a system for the efficient vector computing
    - Apply optimizations considering architecture characteristics

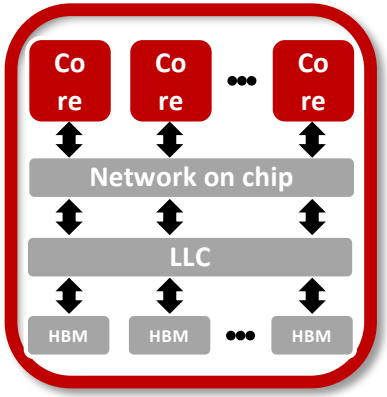
Architecture-aware optimizations

- Multi-leve optimization
  - Node-level
    - Exploit LLC bandwidth
      - to make up for memory bandwidth
    - Expoit HBM2 bandwidth
    - Expoit vector capability
      - Enhance vectorization
      - Reduce the loop overhead
      - Tune the domain decomposition
  - Multi-node level
    - Choose an appropriate path for communications
      - Tune the process mapping





Node-level optimization



- Store reusable data on the LLC
  - Each element of a pressure array can be reused 18 times
    - Store the array in the LLC with a high priority
    - Insert a directive ”#pragma NEC retain(array-name)”

  →Efficient use of the LLC bandwidth

- Loop unrolling
  - Long loops with nested structures in the kernel
  - Use plenty of vector registers to reduce the loop overheads

  →Efficient vector computation

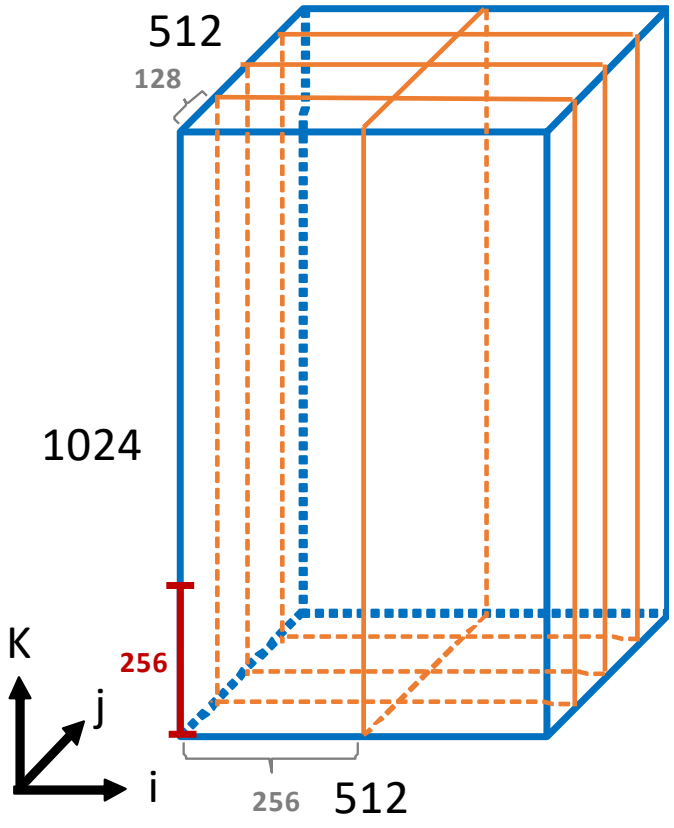
```
1:for(n=0 ; n<nn ; ++n){
2:  gosa = 0.0;
3:  wgosa= 0.0;
4:  for(i=1 ; i<imax-1 ; ++i)
5:#pragma _NEC outerloop_unroll(16)
6:    for(j=1 ; j<jmax-1 ; ++j)
7:#pragma _NEC retain(p)
8:     for(k=1 ; k<kmax-1 ; ++k){
9:       s0 = a[0][i][j][k] * p[i+1][j ][k ]
10:        + a[1][i][j][k] * p[i ][j+1][k ]
```

Node-level optimization (2)

- Domain decomposition
  - k direction >= vector length
    - Keep enough vector length
    - 256+ in the case of SX
  - i direction > j direction
    - Achieve high LLC hit ratio

→ Efficient vector processing

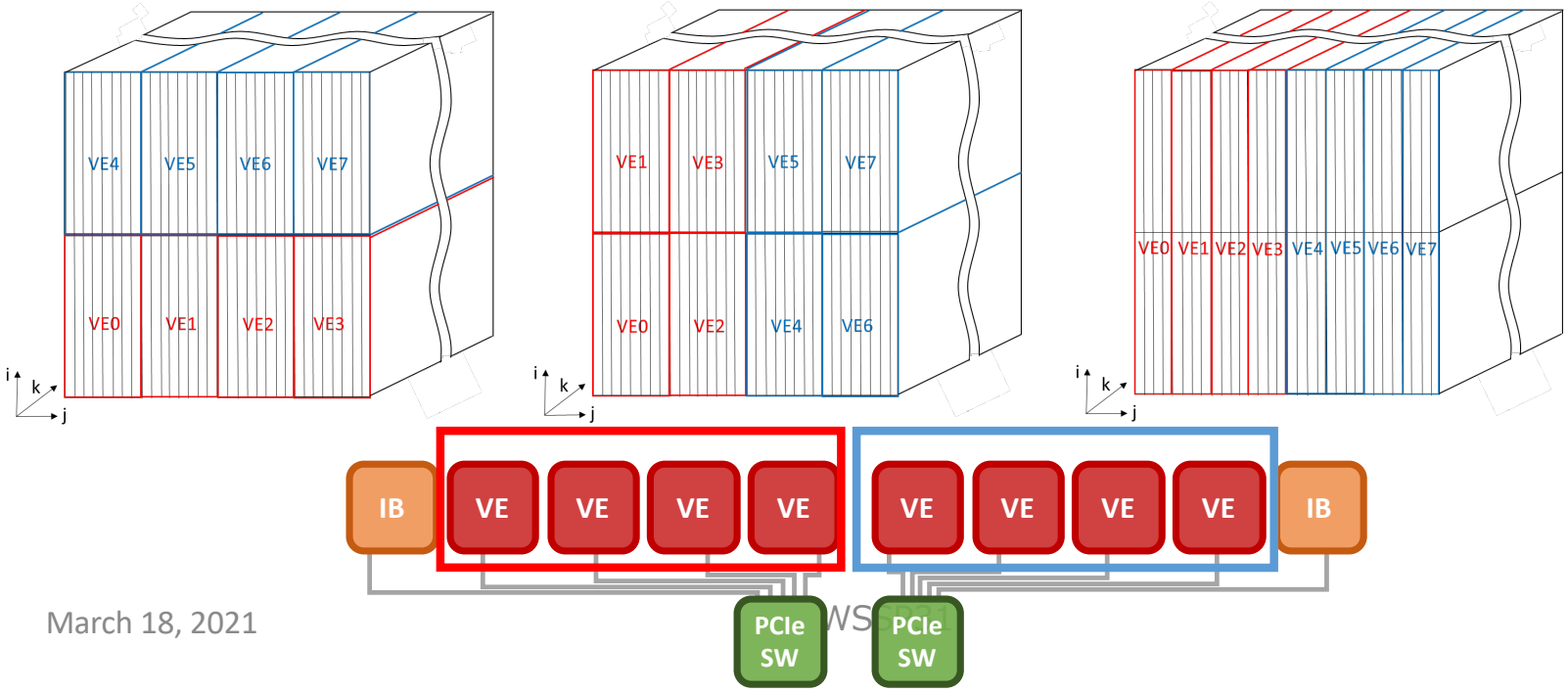
```
1:for(n=0 ; n<nn ; ++n){
2:  gosa = 0.0;
3:  wgosa= 0.0;
4:  for(i=1 ; i<imax-1 ; ++i)
5:#pragma _NEC outerloop_unroll(16)
6:    for(j=1 ; j<jmax-1 ; ++j)
7:#pragma _NEC retain(p)
8:     for(k=1 ; k<kmax-1 ; ++k){
9:       s0 = a[0][i][j][k] * p[i+1][j ][k ]
10:       + a[1][i][j][k] * p[i ][j+1][k ]
```



Decomposition (2, 4, 4)

Multi-node level optimization

- Multi-level bandwidth aware Process mapping
  - Processes to calculate adjacent domains are assigned to the cores in the same node
  - Amounts of data transfer among different nodes in different PCIe switches are reduced
  - Processes to communicate with different PCIe switch are distributed in a node
    - Load of data transfers should be equally distributed
    - Multiple VEs in a PCIe switch can communicate different PCIe switches in the case of SX



Multi-VE/VH level optimization(2)

- Procedure
  - Measurements of multi-level bandwidth of a system
    - Bandwidth is measured using a benchmark program
    - This step needs to be done when the system is installed
  - Analysis of communication of an application
    - Communication are analyzed as a preliminary setup
      - patterns, amounts, orders, and so on
  - Mapping processes considering an application and a system

Experimental Environment s

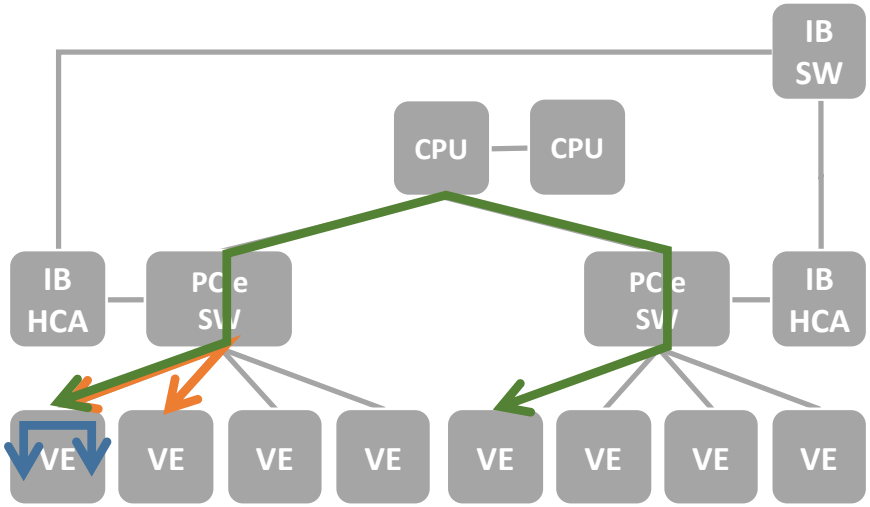
- Systems

System	Model	Host	Accelerator
Aurora1	A300-2	Xeon Gold 6126 x1	VE Type 10B x1
Aurora2	A300-2	Xeon Gold 6126 x1	VE Type 10B x2
Aurora8	A300-8	Xeon Gold 6126 x2	VE Type 10B x8
Xeon		Xeon Gold 6126 x1	-
V100		Xeon Gold 6126 x1	V100 x1

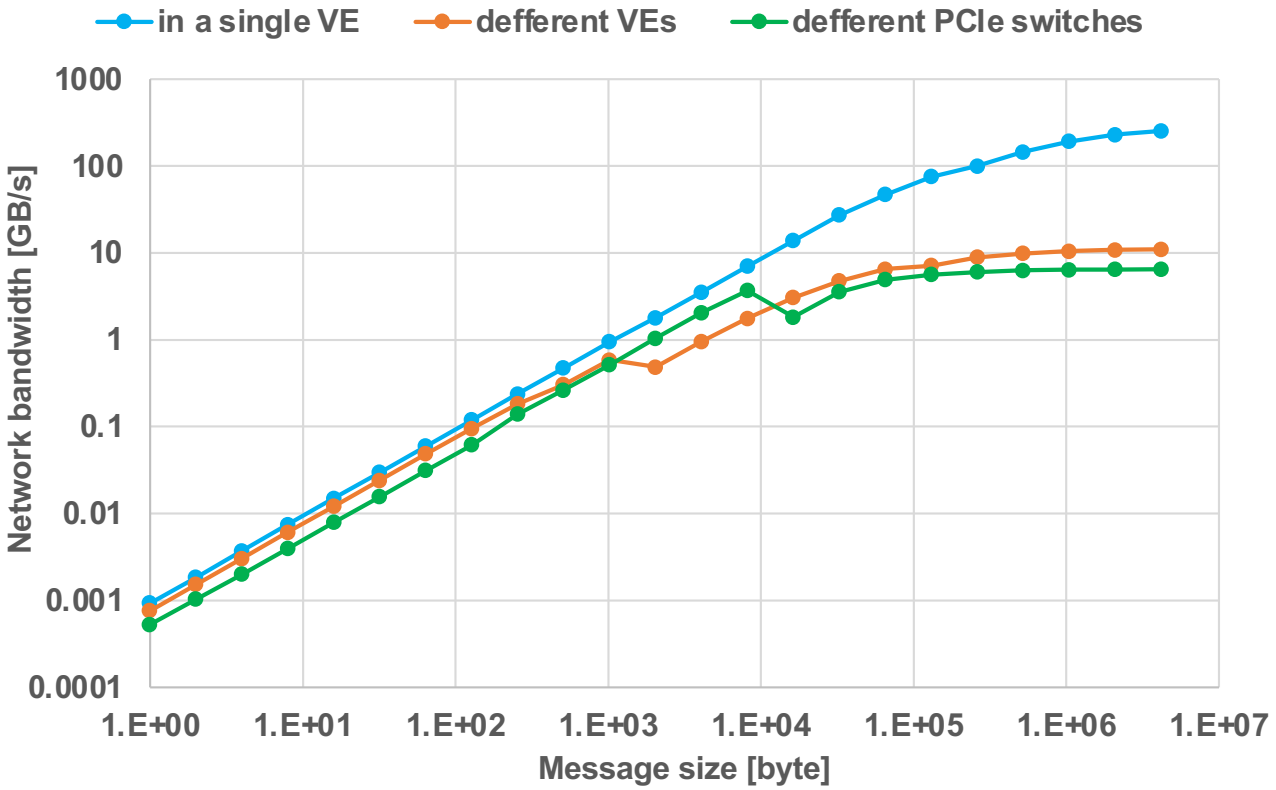
- Benchmark
  - Aurora, Xeon: Himeno benchmark MPI - C language version
  - V100: Himeno benchmark OpenACC - Fortran version
- Evaluation
  - Performance
  - Scalability
  - Power efficiency

	NEC VE Type 10B	Intel Xeon Gold 6126	NVIDIA Tesla V100
Performance (SP)	4.30 Tflop/s	883.2 Gflop/s	14 Tflop/s
Memory bandwidth	1.22 TB/s	128 GB/s	900 GB/s
LLC bandwidth	2.66 TB/s	N/A	N/A

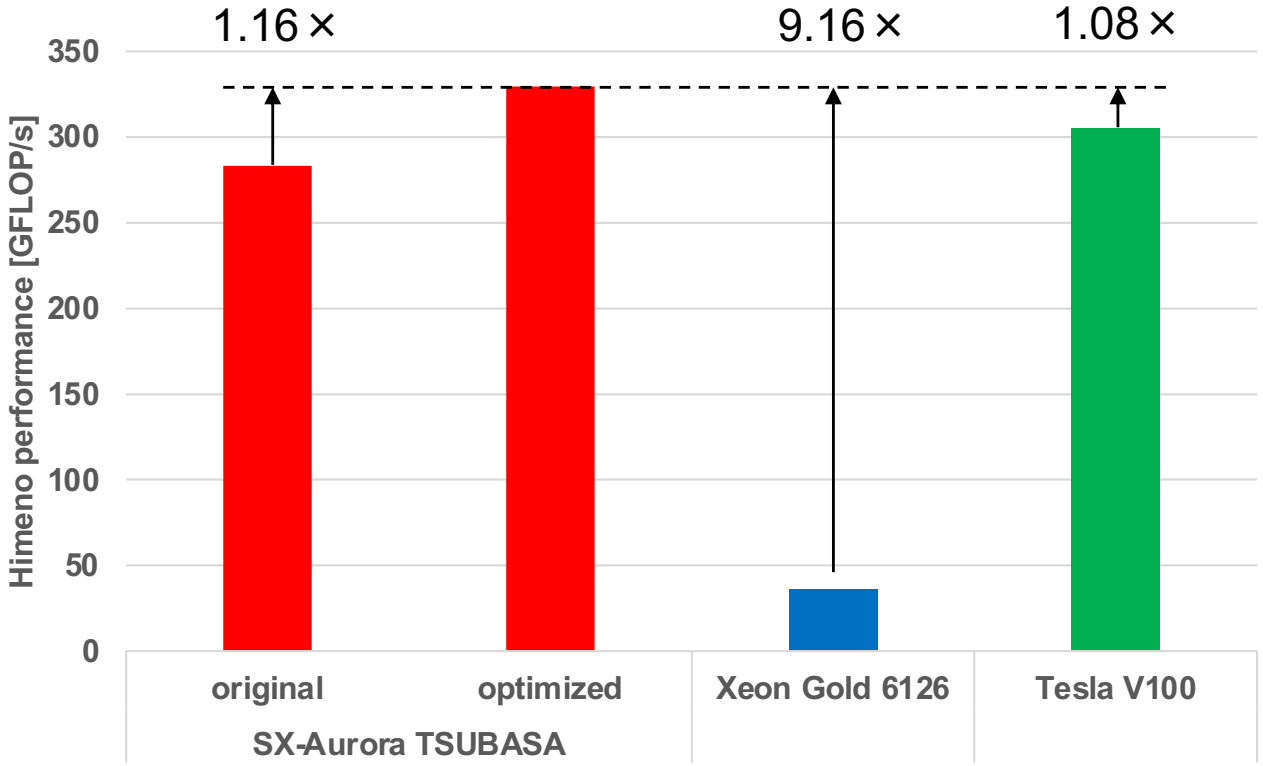
Bandwidth of SX-Aurora TSUBASA



- OSU benchmark

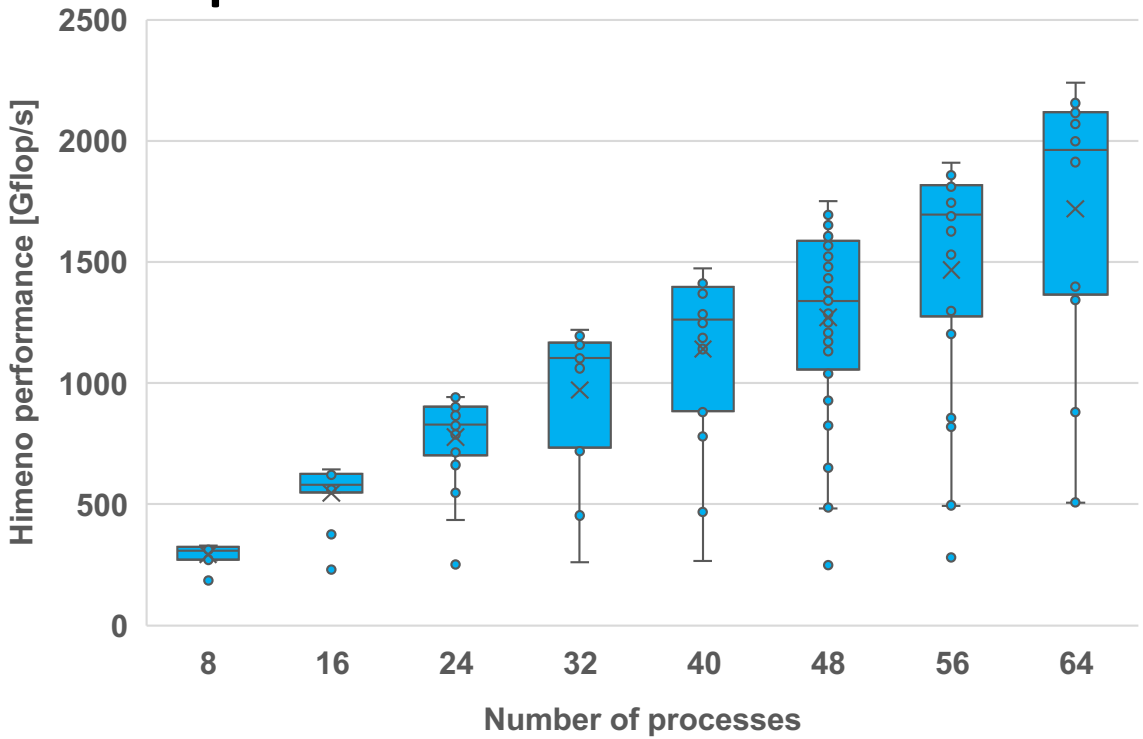


Single VE performance



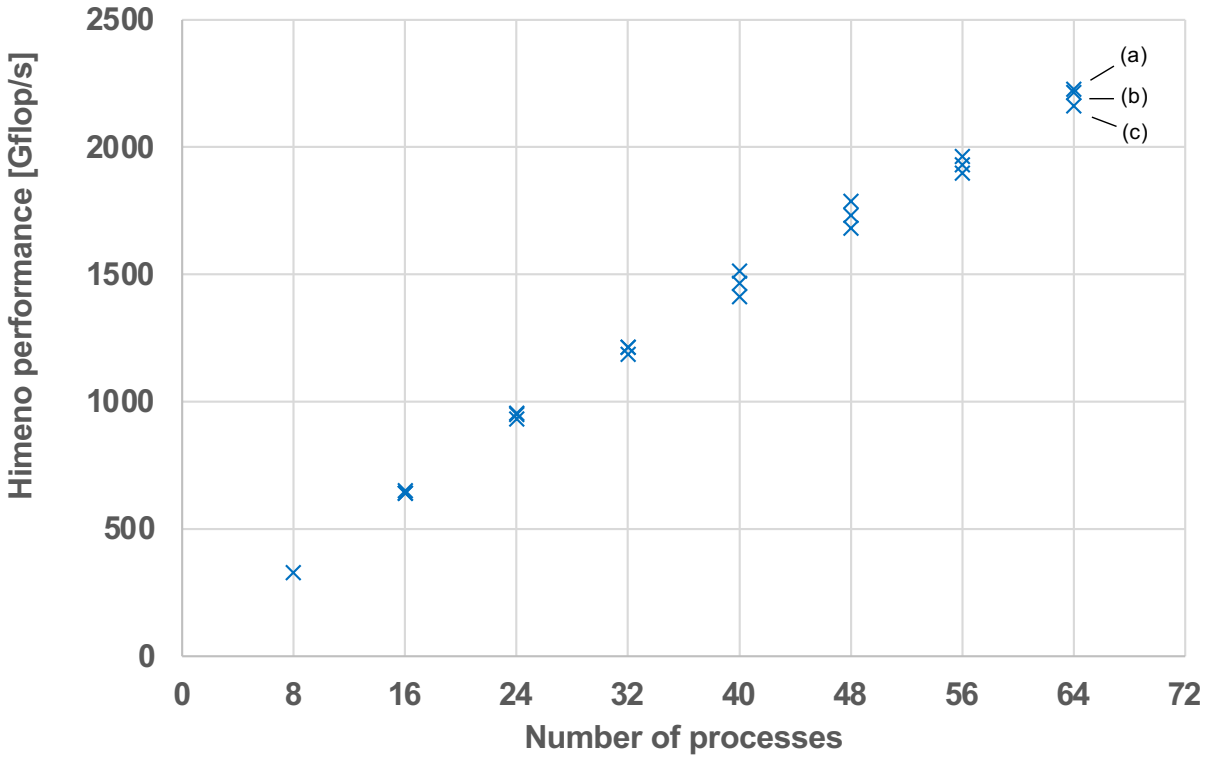
- 16% improvement over the original code
- High efficiency
  - SX-Aurora TSUBASA original: 6.6%, optimized: 7.7%, Xeon: 4.1%, V100 2.2%

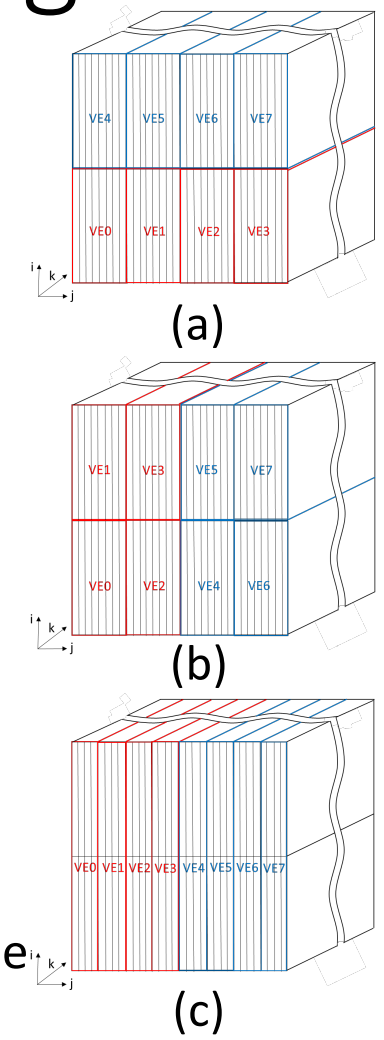
Multiple VEs: domain decomposition



- The domain decomposition impacts on the large number of processes
  - Good parameters in most cases
    - K direction size should be large
    - J direction size < i direction size

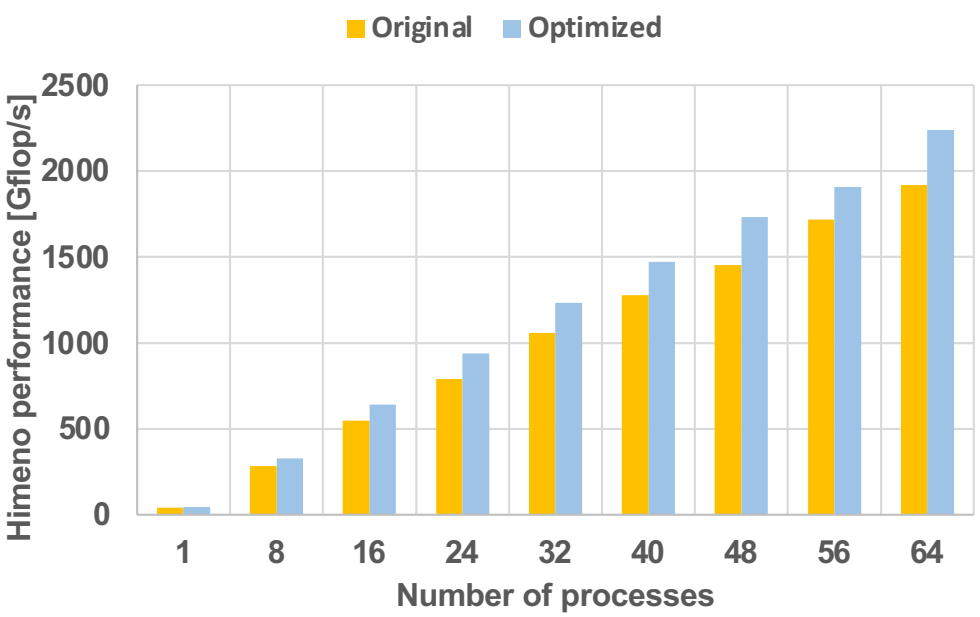
Multiple VEs: process mapping

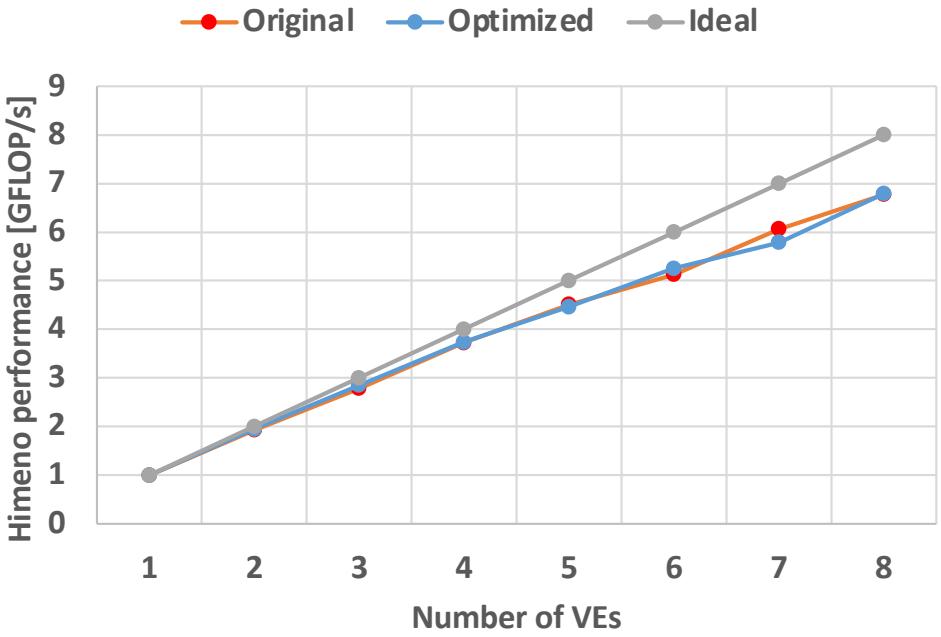




- Process mapping is effective more than 40 processes
  - The data transfer between different PCIe switches affects the performance
  - About 50~100 Gflop/s performance differences

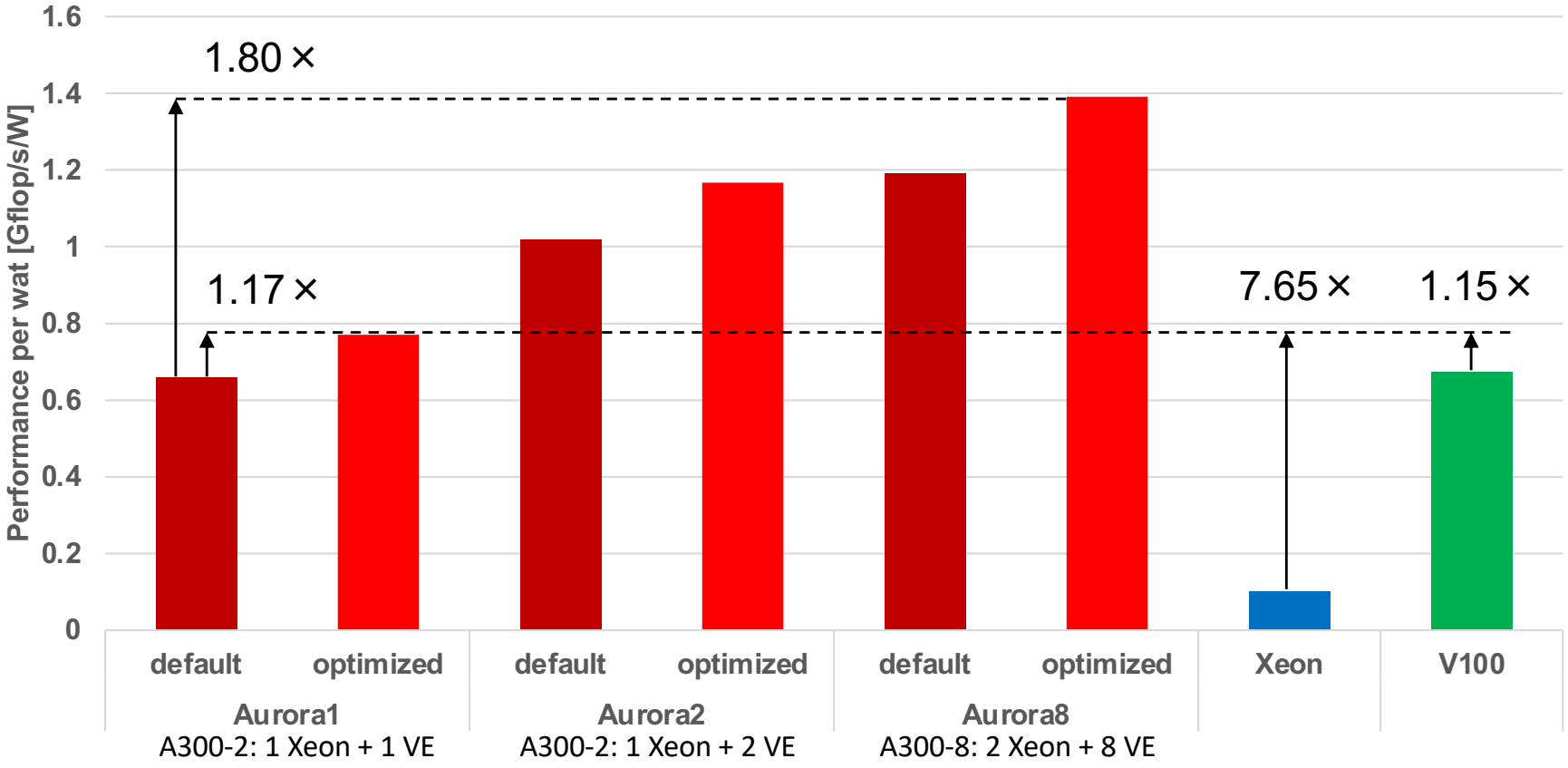
Multiple VEs: All optimizations





- Higher performance over the original version
  - 6.79 x performance improvement in 8 VEs
  - 84.9% parallel efficiency in 8 VEs

Performance per Watt



- The performance per power improves when the large number of VEs
  - Optimization do not affects the power consumpition
  - The power consumpiton does not increase much even when the number of VEs increases

Conclusions

- Conclusions
  - SX-Aurora TSUBASA has achieved high performance of the himeno benchmark by the optimizations
    - Multi-levels of bandwidth is exploited
    - Loop unrolling and tuning of the domain decomposition are relatively effective
    - Higher performance than Xeon and V100
  - Performance per power also improves by optimizations

- Future Work
  - More detailed analysis of the process mapping
    - Cooperation of the domain decomposition and the process mapping
  - Evaluate and analyze on a large system