# Decoupling Architecture for All-to-all Computation

## SACLA XFEL and the K Computer

Atsushi Hori

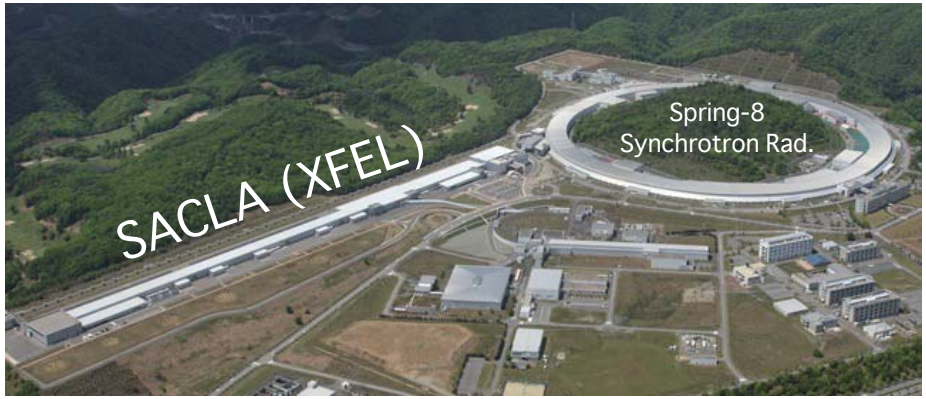
K. Yoshinaga, A. Tokuhisa, Y. Joti, K. Okada,
T. Sugimoto, M. Yamaga, T. Hatsui,
M. Yabashi, Y. Sugita, Y. Ishikawa, and N. Go

# Outline

- SACLA XFEL and the K Computer
- Analyzing Diffraction Images
- Load Balancing and Minimizing I/O
- Decoupling Architecture
- Evaluation
- Summary

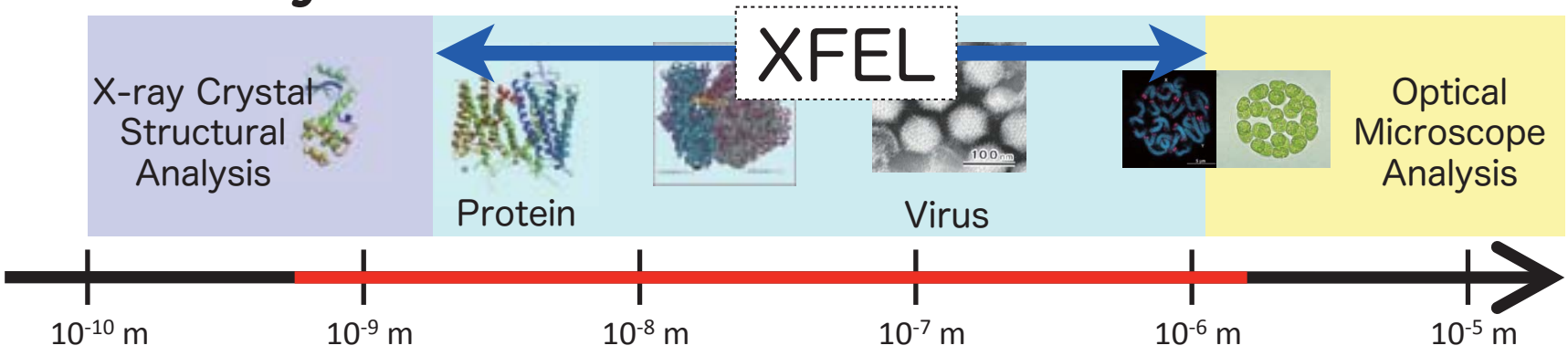
# SACLA and the K Computer

- Data Acquisition: SACLA (XFEL Facility)
  XFEL: X-ray Free Electron Laser
- Data Processing: the K computer




WAN
(80Km)

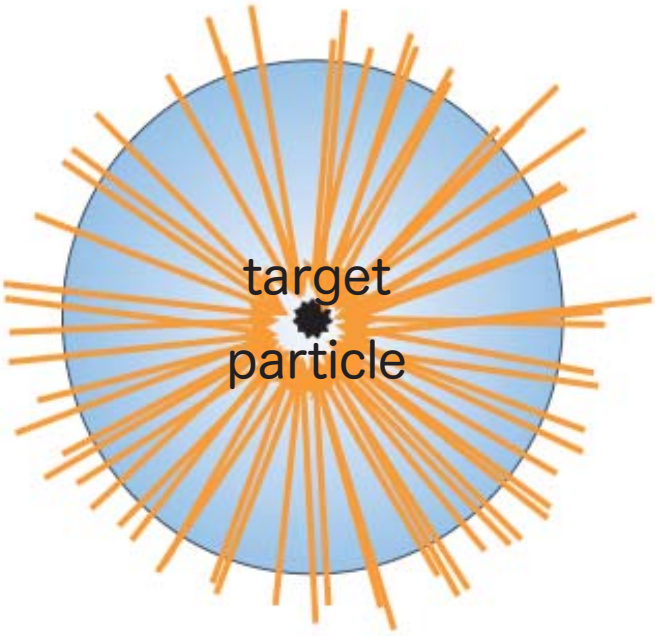

- To Analyze 3D Structure of Particles



# Analyzing Diffraction Images (1/2)

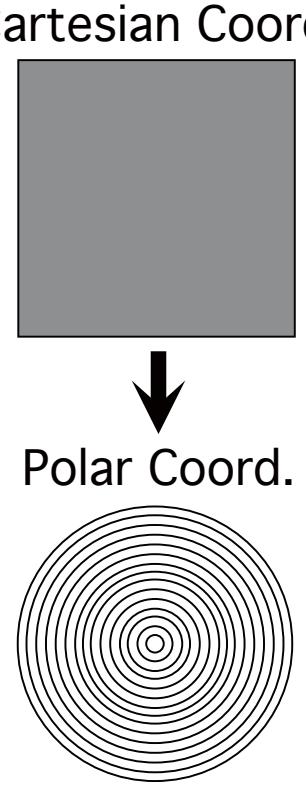
- Orientations of target particles are uncontrollable
  - Correlation (FFT) is the clue of the orientation
    => Clustering
  - All-to-all computation is needed $O(N^2)$ !
- Diffraction image contains quantum noise
  - 100 images must be averaged.

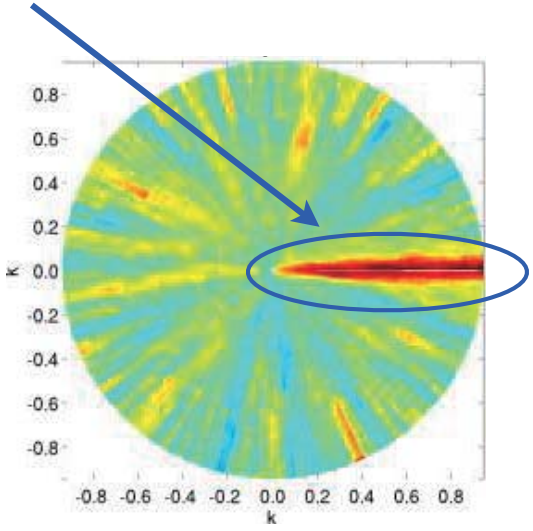
➡ 1 million images must be shot !!



# Analyzing Diffraction Images (2/2)

- 2-D Cartesian coordinate image is converted to Polar coordinate image (coaxial rings)
- Each ring is FFTed
- Two images are compared (correlation), ring by ring
- Finally, resulting “a correlation line” indicating they are close enough




High-speed classification of coherent X-ray diffraction patterns on the K computer for high-resolution single biomolecule imaging (A. Tokuhisa, et al.), In Journal of Synchrotron Radiation, volume 20, 2013.

# Fact Sheet

- Data Acquisition (SACLA)
  - max. 30 images/sec (depending on the kind of target particle)
  - 20MB/image (3Å resolution) (depending on resolution)
  - 1 million images for one particle analysis
    - quantum noise and all possible orientations

  ➡ yielding 20TB in 10 hours !
- Data Transfer (from SACLA to “K”)
  - *Gfarm* copy tool (*gfpcopy*) takes 20 hours to copy 20TB data
- Data Processing (“K”)
  - $O(N^2)$ , but can be reduced to $O(K \cdot N)$

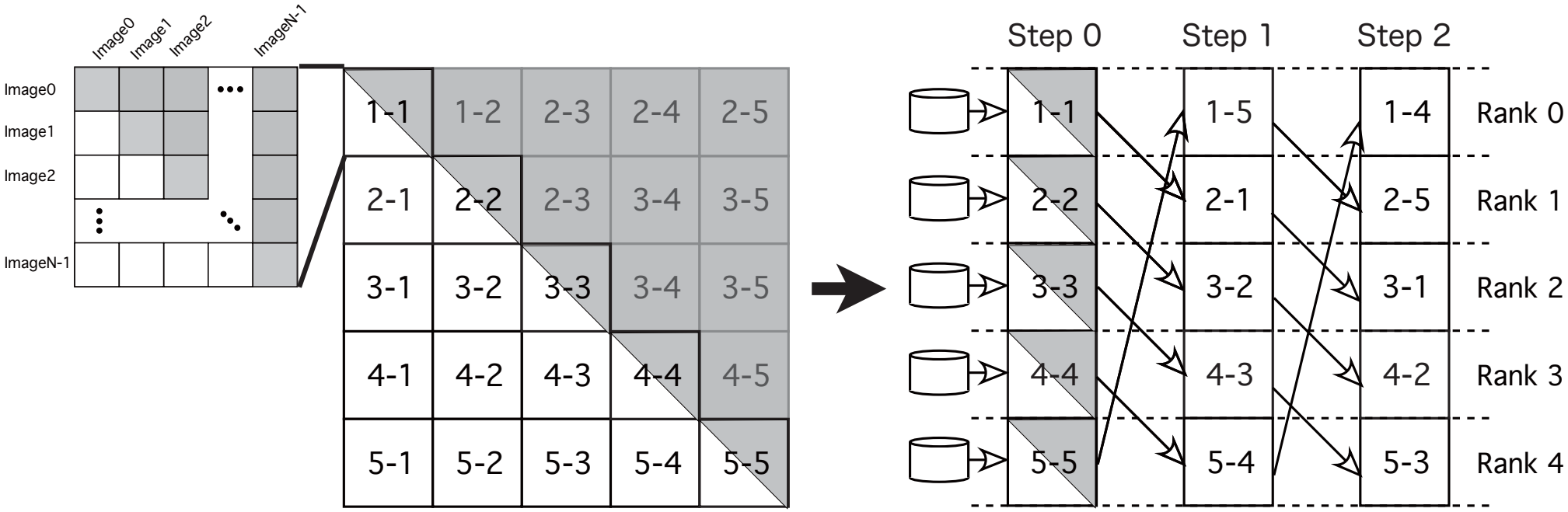
    $K$ is the number of clustering groups

# Load Balancing and Minimizing I/O

- Load must be balanced
- Each data file must be read *only once*

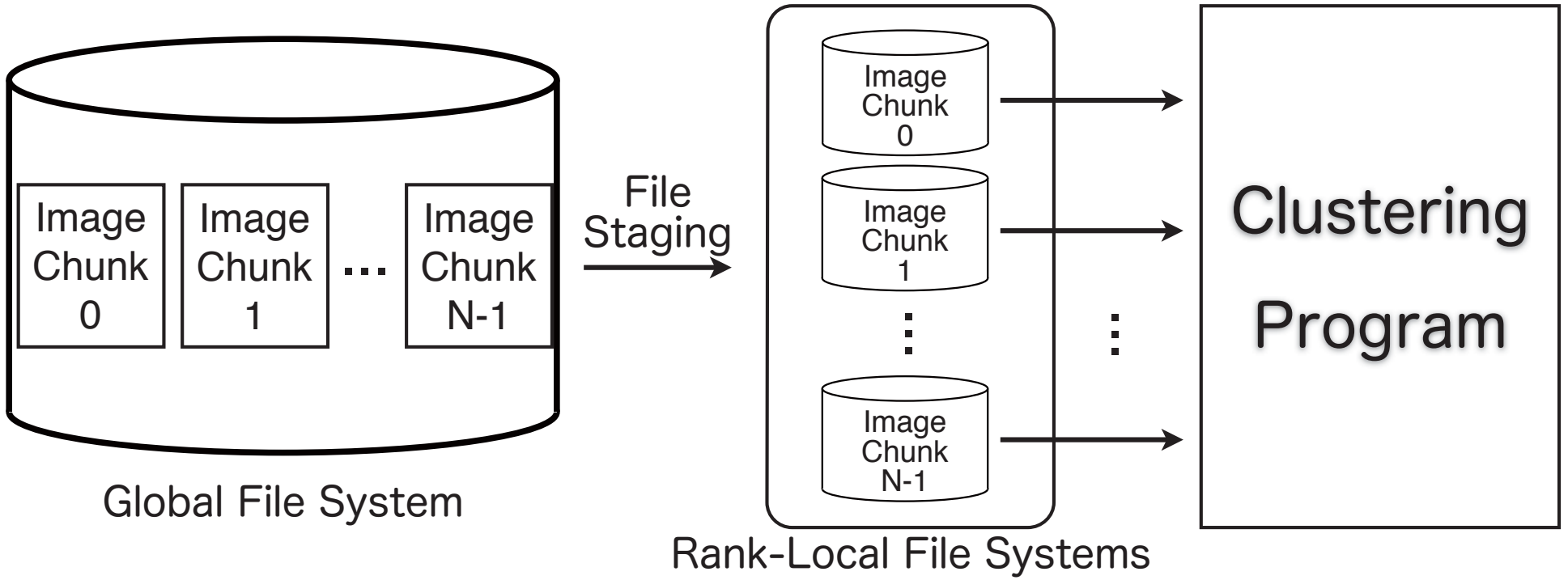
➡ Read data are passed to neighbors at every step

- #steps is *(N+1)/2* (*N* is #chunks or #nodes)



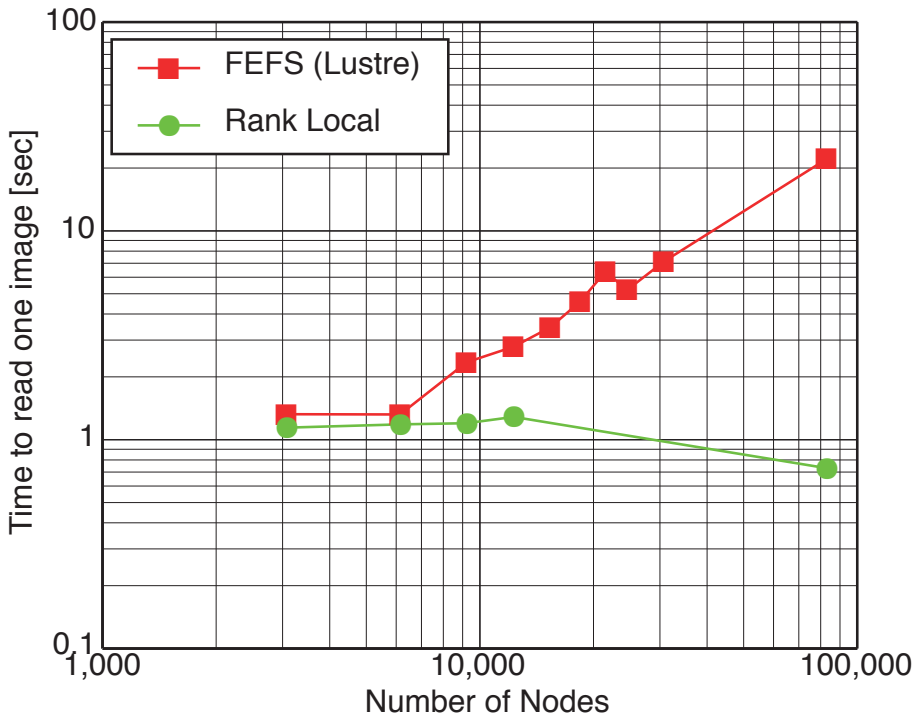
# I/O Optimization

- Parallel file system can be a bottleneck
- Maximizing parallelism up to #nodes
- ➡ Utilizing the K's file system
  - File staging system, and
  - "Rank-Local" file system



# I/O Performance

- Computation time
  - 5Å 0.06 Sec/2images
  - 3Å 0.45 Sec/2images
- Communication time
  - Hidden
- Read Time




- Computation time dominates (“K”, 80K nodes)
  - 5Å 1 hour
  - 3Å 24 hours

# Decoupling Architecture

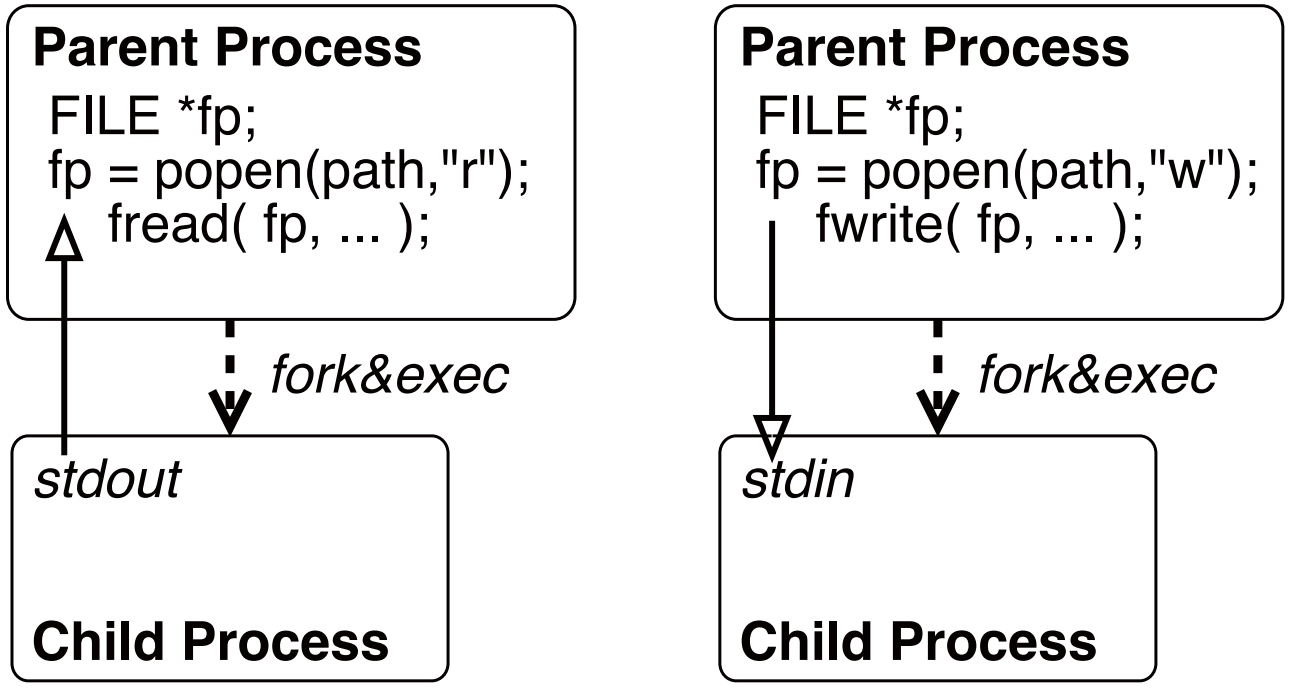
# Decoupling Architecture

- Motivation
  - Generalizing “all-to-all computation”
    - Lowering the threshold to use the K computer
- Decoupling Architecture
  - Decoupling the kernel code and the other system programing staff
    - by using the glibc popen() function

Hadoop Streaming

# popen

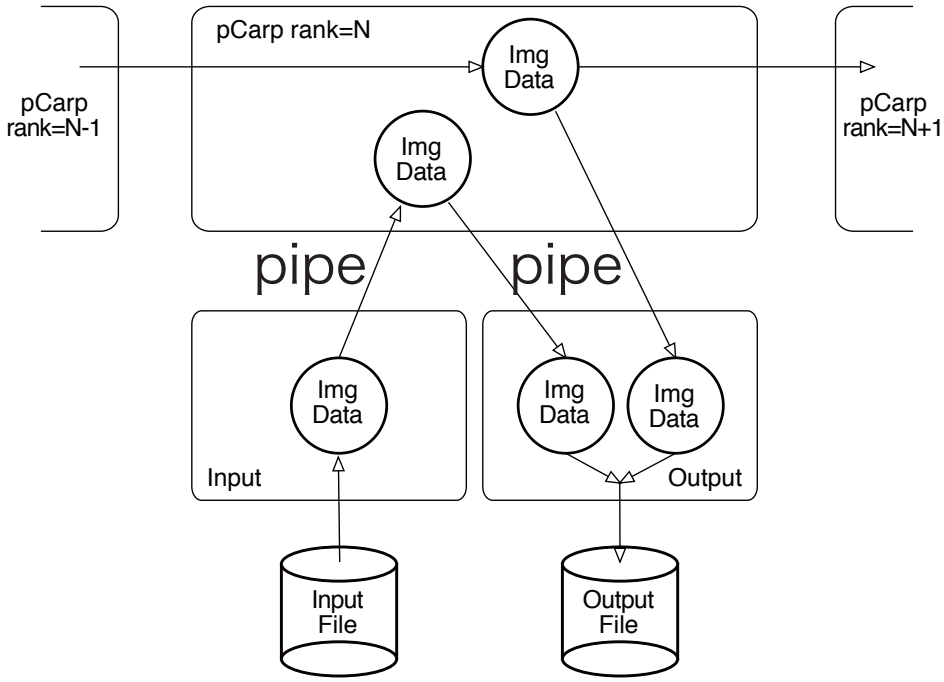
- *popen* creates a subprocess, and
- returns a file pointer of a pipe which is connected with the stdout or stdin of the subprocess




**(a) popen( "r" )** **(b) popen( "w" )**

# Carp

- pCarp
  - Parallel version for production run
- sCarp
  - Sequential version for kernel development and debugging

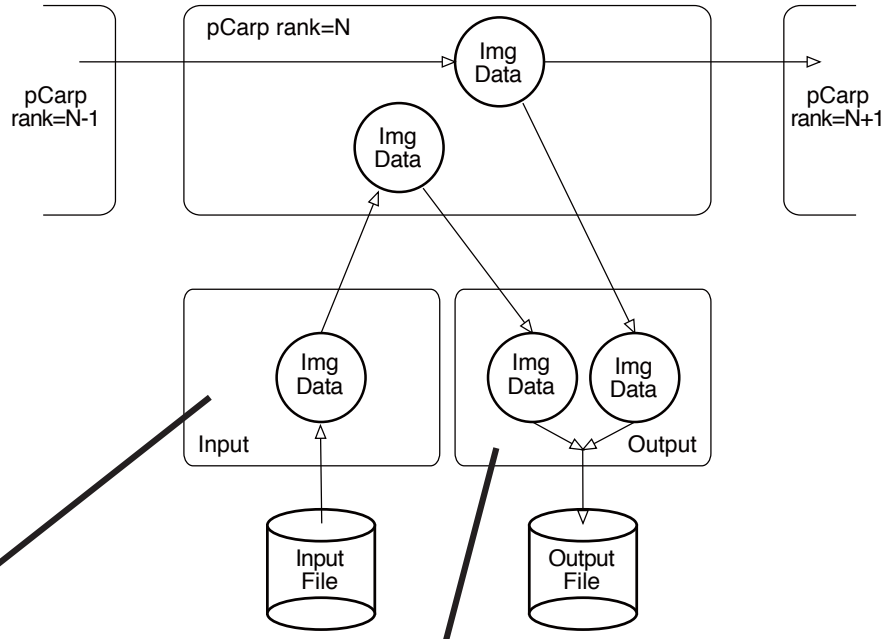



```
$ mpiexec pcarp args input_prog args output_prog args
or
$ scarp args input_prog args output_prog args
```

# Code Skeleton

Users do not care about

- Programming Language
- MPI
- Load Balance
- I/O



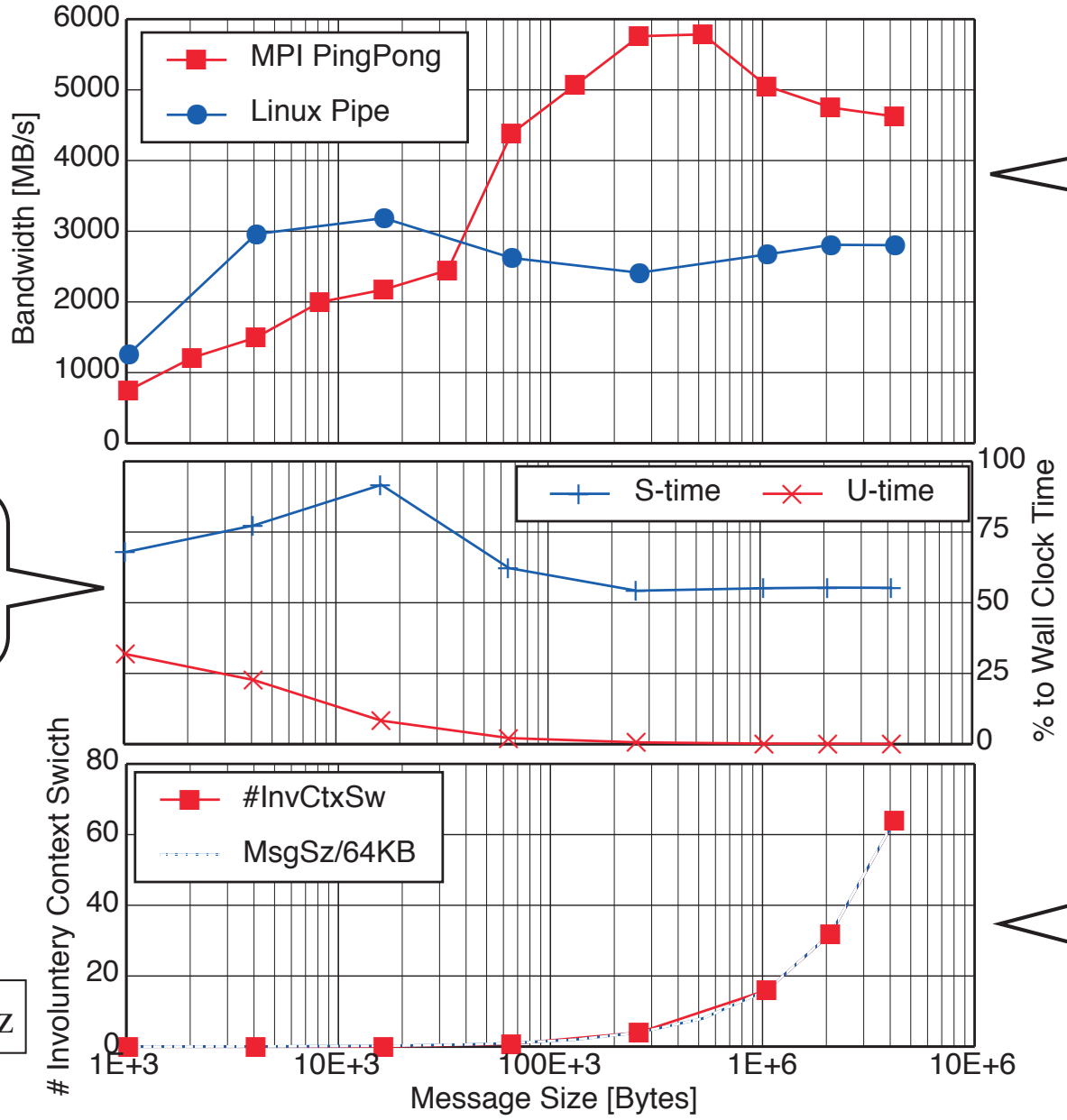

```
main( argc, argv ) {
    data = OpenReadDataFile( argc, argv );
    CarpWrite( data, size );
}
```

```
main( argc, argv ) {
    for( ... ) {
        data0 = CarpRead();
        data1 = CarpRead();
        result = kernel_code( data0, data1 );
        OutputData( result );
    }
}
```

# pCarp Performance

- Original Program
  - 240 Sec.
- pCarp + Sequential Programs
  - 430 Sec. (+190 Sec.)
    - Pipe Transfer: 159 GBytes
    - Pipe BW: 1 GB/s
    - Most of the overhead comes from the transfer via the pipe (160 Sec.) !!

# Pipe Performance




Pipe BW is BAD !!

Half CPU utilization !!

Ctx. Sw. is the problem

E5-2650 v2, 2.60 GHz

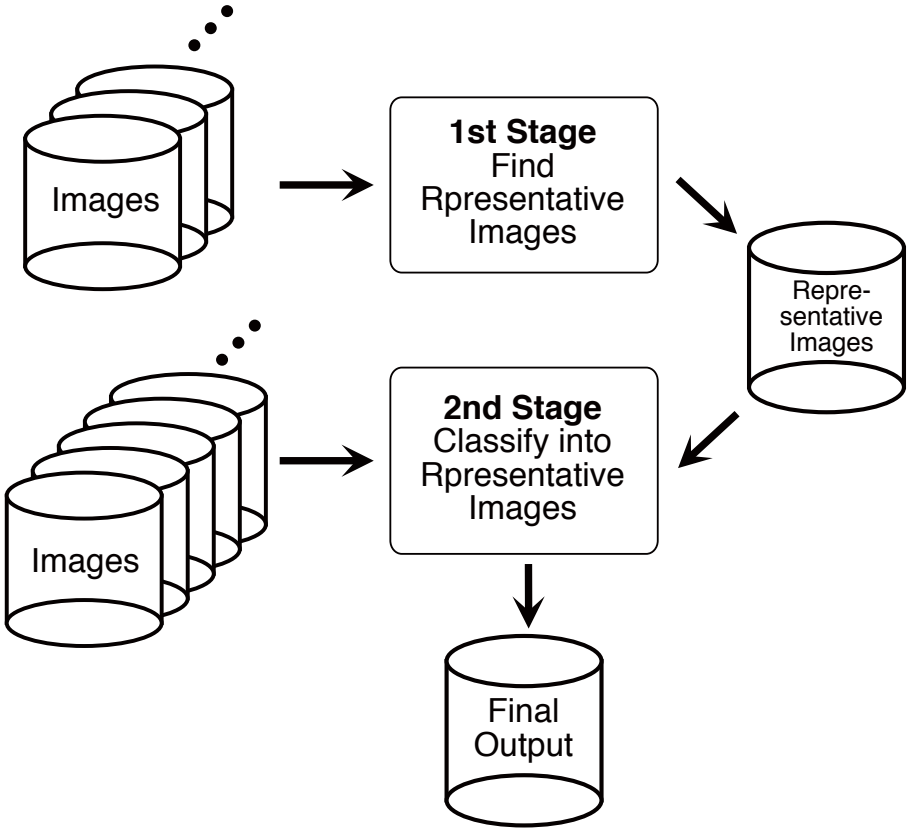
# Summary

- Decoupling Architecture
  - Users do not to worry about the parallel programing (I/O, communication, load-balancing, ...)
  - Users can write their programs in any programing languages

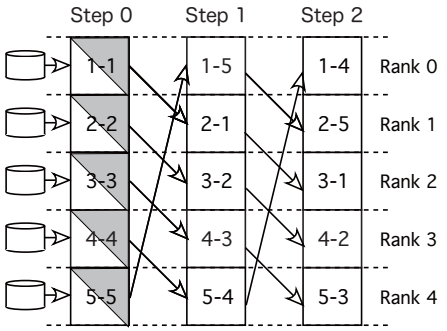
# Future Work

- [ps]Carp improvement - in the future
  - Increasing the pipe buffer size in the Linux kernel
    - fcntl( fd, F_SETPIPE_SZ )
      - Linux 2.6.35～, or
  - Developing new “popen” implementation to improve the bandwidth
- Having applications other than XFEL

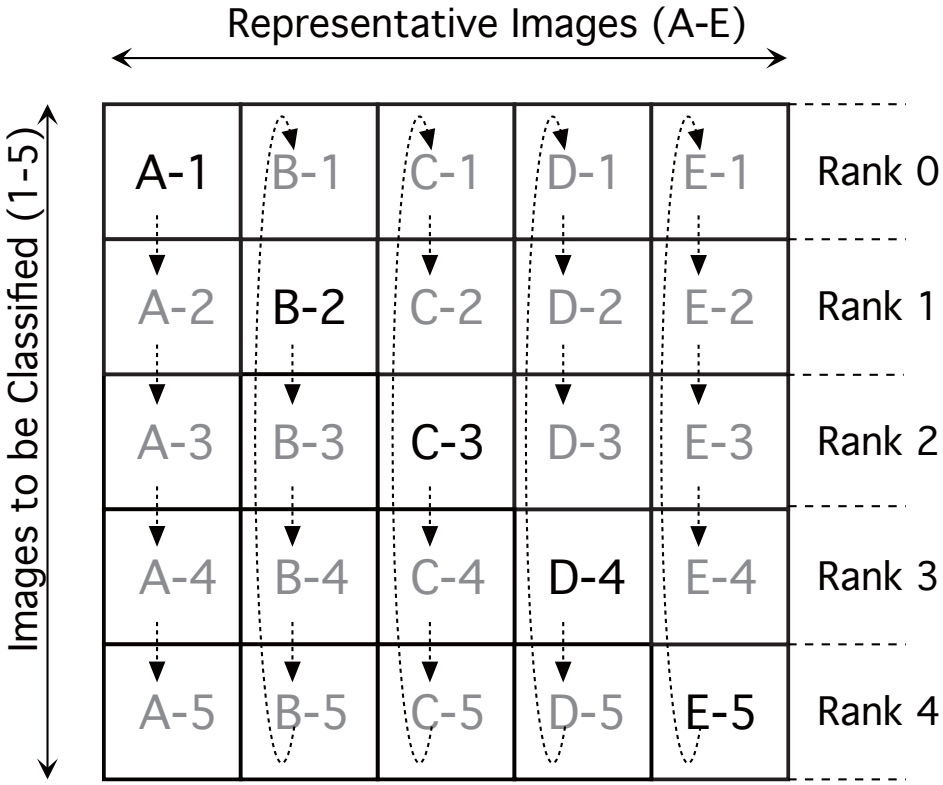
# Stage1 and Stage2




## 1st Stage




## 2nd Stage



# XFEL in the World

http://en.wikipedia.org/wiki/XFEL#X-ray_FELs

- FLASH (Free-electron -LASer in Hamburg)
- Linac Coherent Light Source (LCLS) at the SLAC National Accelerator Laboratory
- European x-ray free electron laser
- Paul Scherrer Institute (Switzerland)
- SACLA at the RIKEN Harima Institute in Japan