UNIVERSITY *of* DELAWARE

Challenges in Big Data Computing on HPC Platforms

Michela Taufer
Computer and Information Sciences
University of Delaware
Newark, Delaware, USA

Acknowledgements


Travis J.


Boyu Z.


Trilce E.


Adam L.


Silvia C.


Ewa D.


Michel C.


Dong A.


Don L.


Rafael D.


Stephen H.

Sponsors:

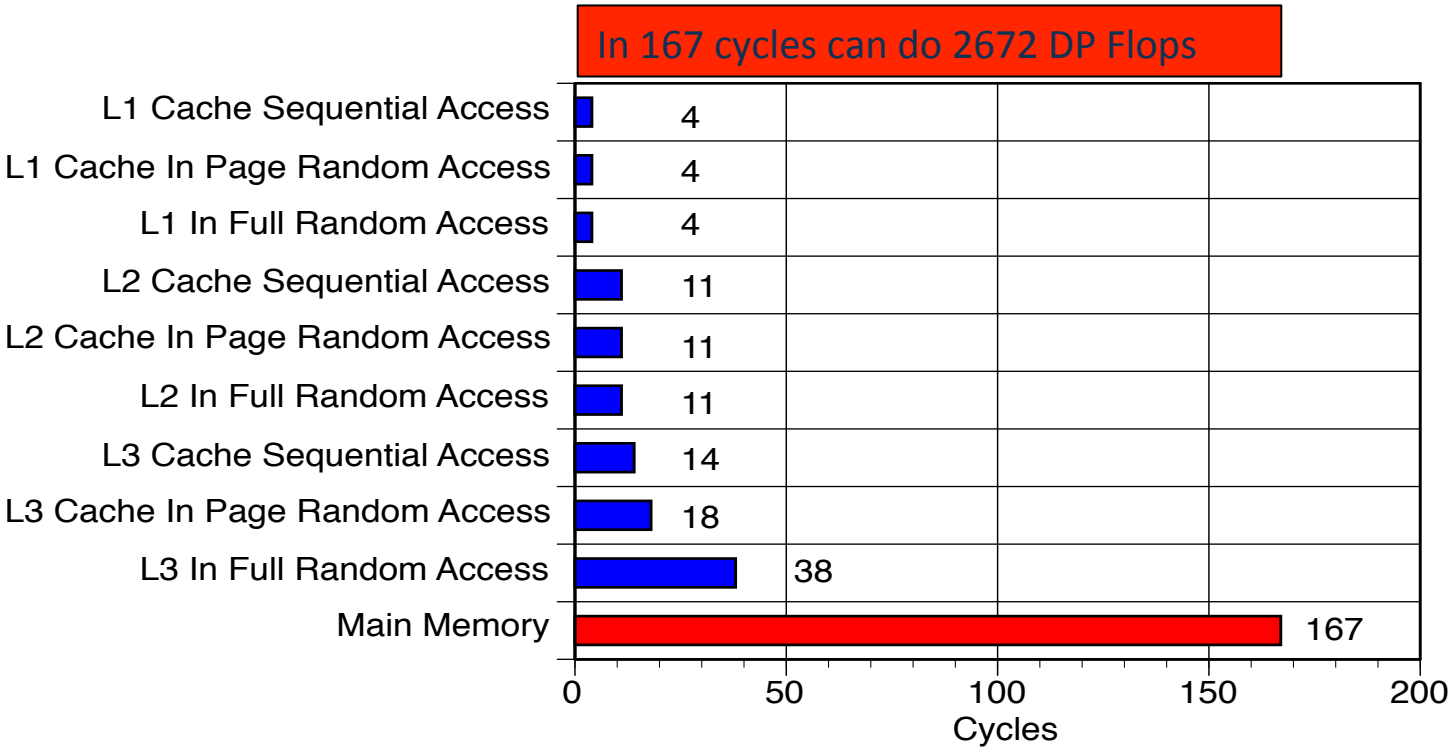
XSEDE
Extreme Science and Engineering Discovery Environment

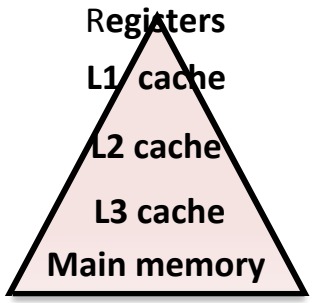
NSF

UD

The Cost of Data Movement

- Today floating point operations are inexpensive



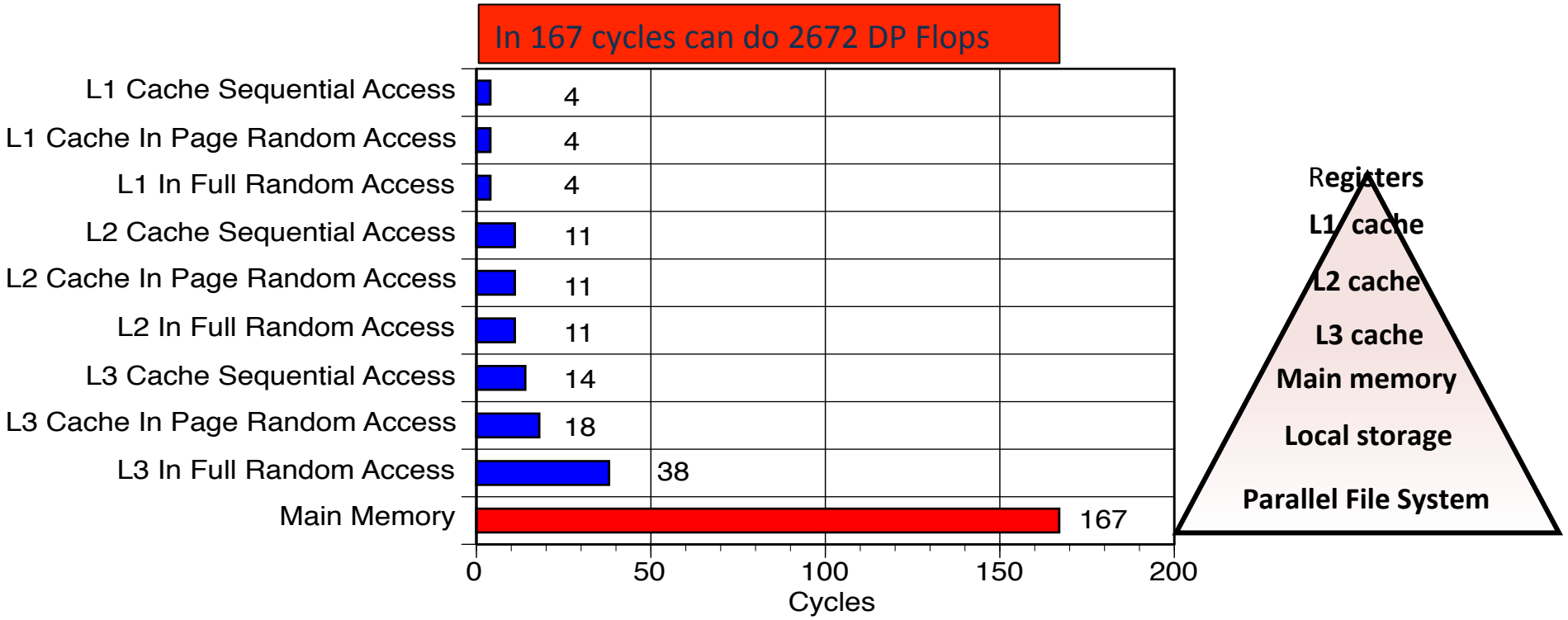


- Data movement is very expensive

Courtesy of Jack Dongarra, UTK and ORNL

The Cost of Data Movement

- Today floating point operations are inexpensive

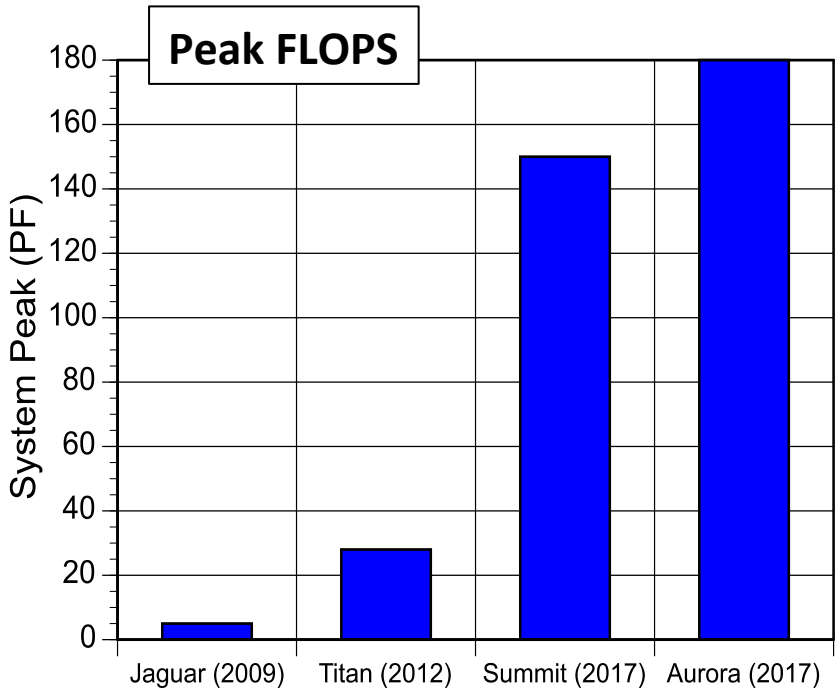


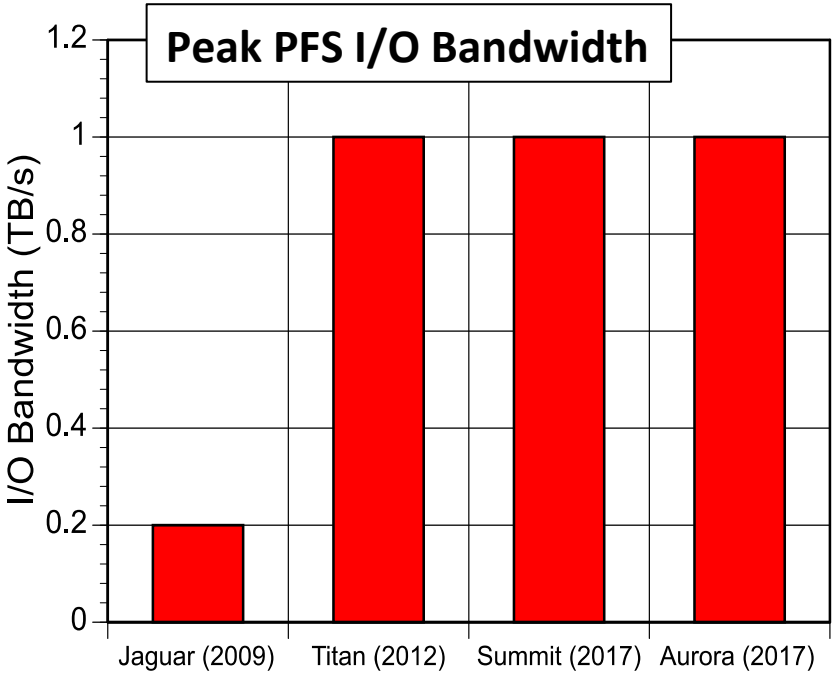
- Data movement is very expensive

Courtesy of Jack Dongarra, UTK and ORNL

The Cost of Data Movement

- Floating point operations will further increase





- Speed to move data down the memory hierarchy is stagnant

Courtesy of Lucy Nowell, DoE

Perspective

The scientist:

"Storage technologies are advancing […] and it is really not clear at all [to me] that especially distributed storage platforms would not be able to handle […] petabyte data sets"

Anonymous Feedback

The computer architect:

"[…] there will be burst buffers on the DOE machines which will give applications much faster I/O […]"

Anonymous Feedback

Burst Buffers

Many have heard about it,

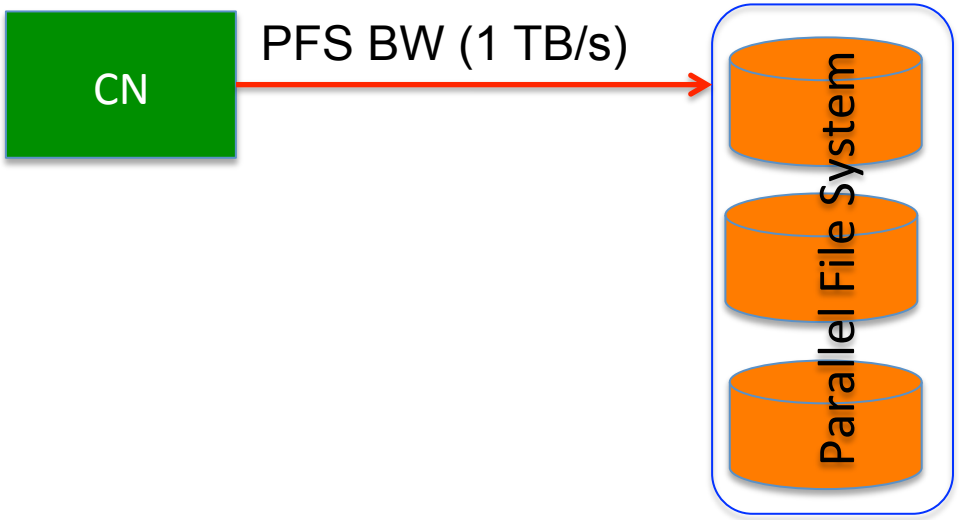
few have seen real machines with it,

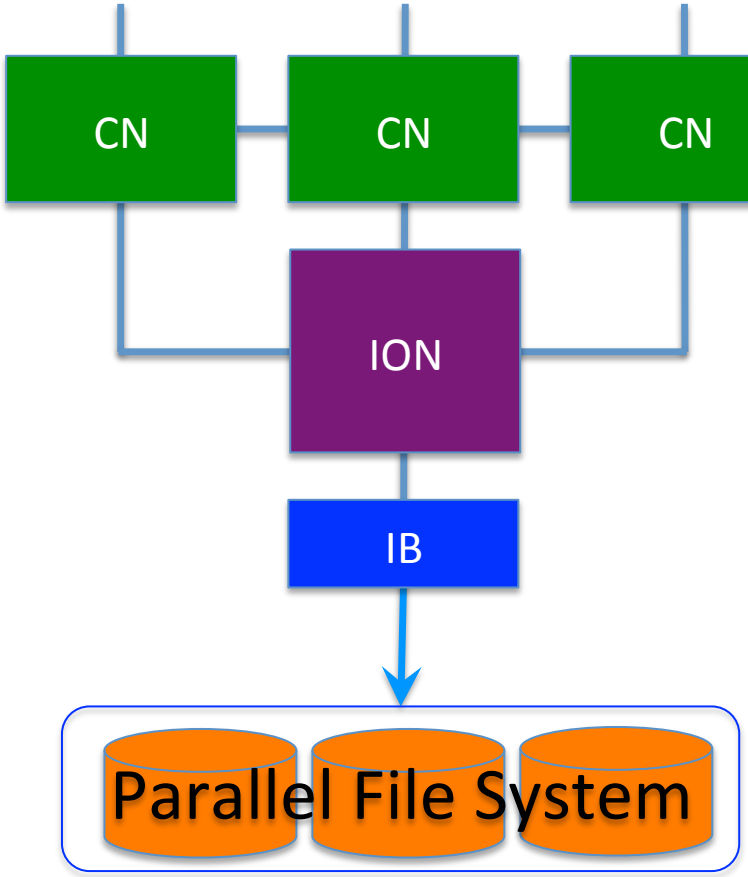
even fewer have ran applications on those machines …

Challenges

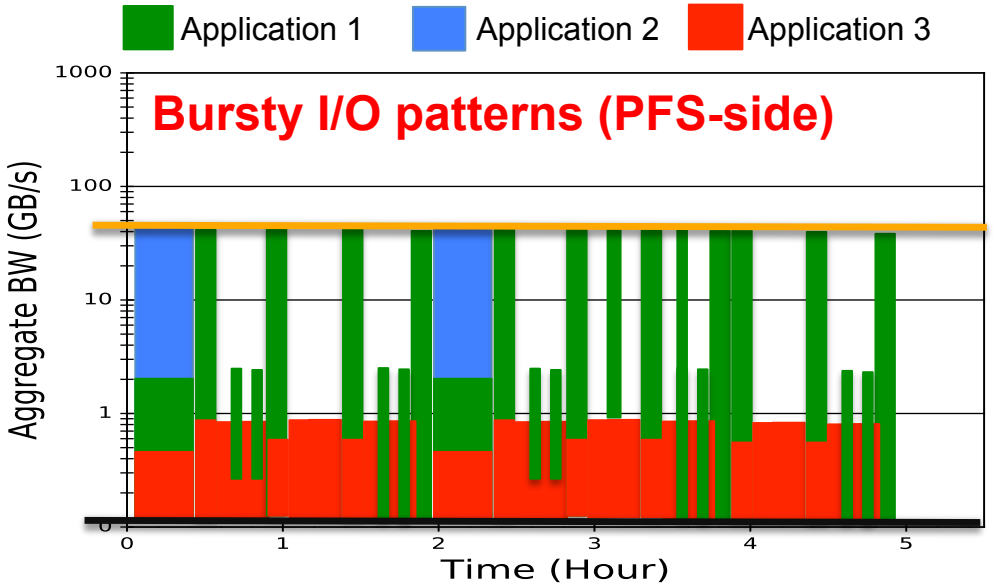
- Burst Buffers are not the magic I/O silver bullet
  - I/O contention still a problem if we exceed the burst buffer capability
  - Burst buffers improve offloading bandwidth but do **NOT** help uploading data from storage for analysis and visualization

Traditional System



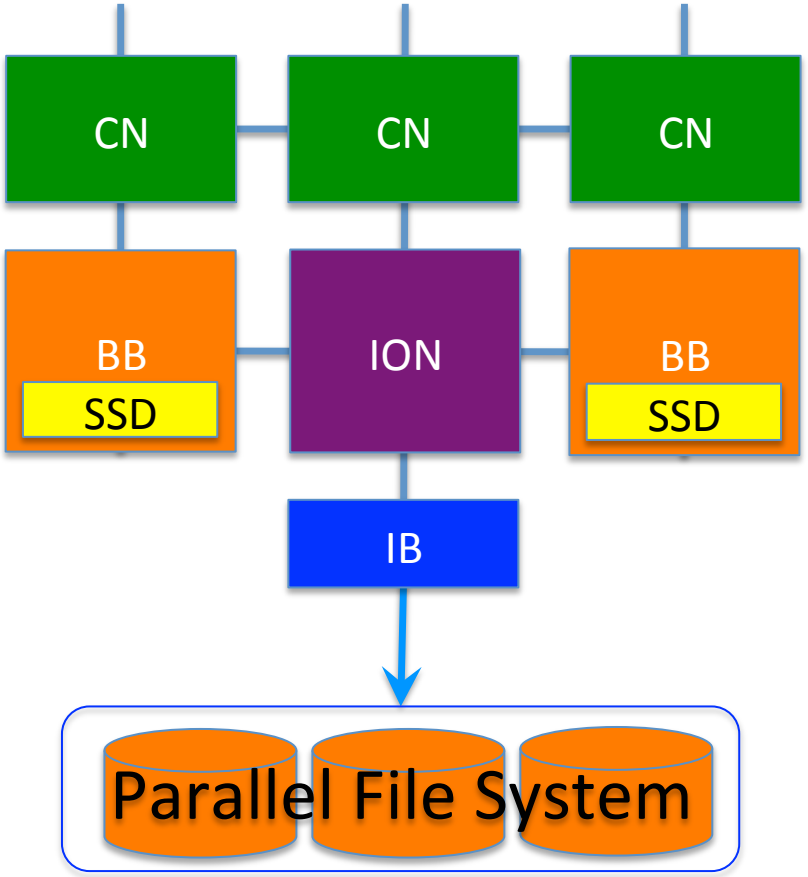


Based on: *http://www.nersc.gov/users/computational-systems/cori/burst-buffer/burst-buffer/*

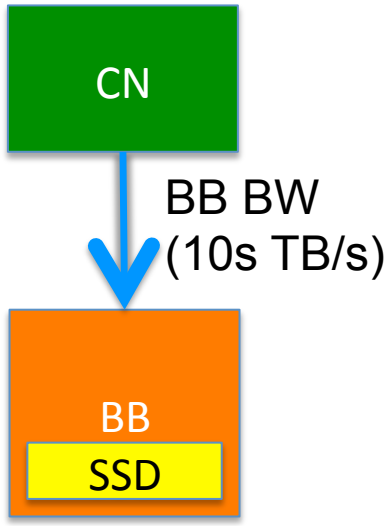


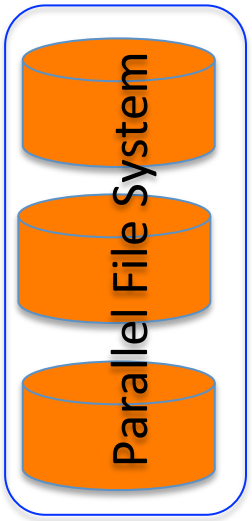
Based on: Liu, N, Cope, J, Carns, P, Carothers, C, Ross, R, Grider, G, Crume, A, Maltzahn, C . "On the Role of Burst Buffers in Leadership-class Storage Systems" MSST/SNAPI 2012

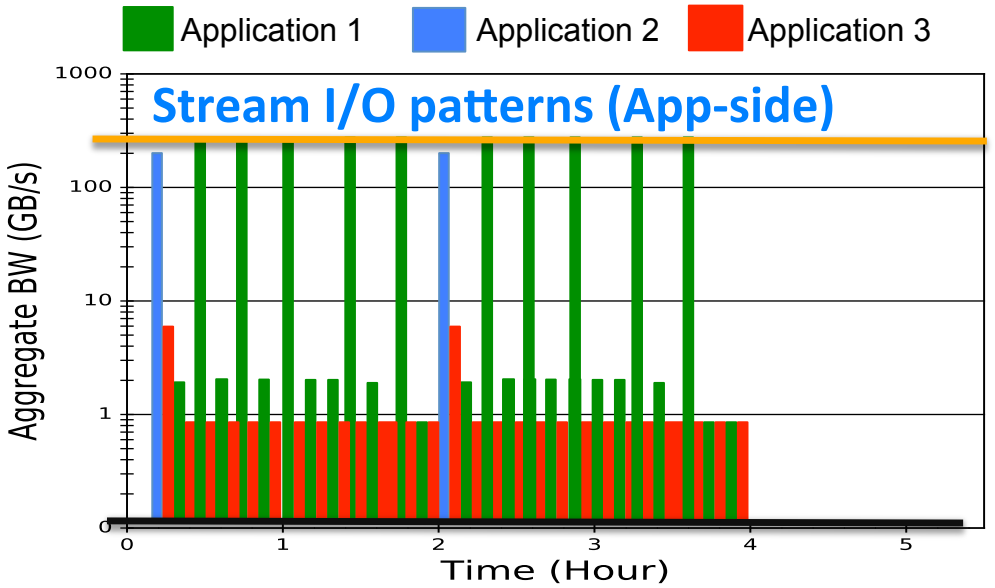
Burst Buffer System



Based on: *http://www.nersc.gov/users/computational-systems/cori/burst-buffer/burst-buffer/*

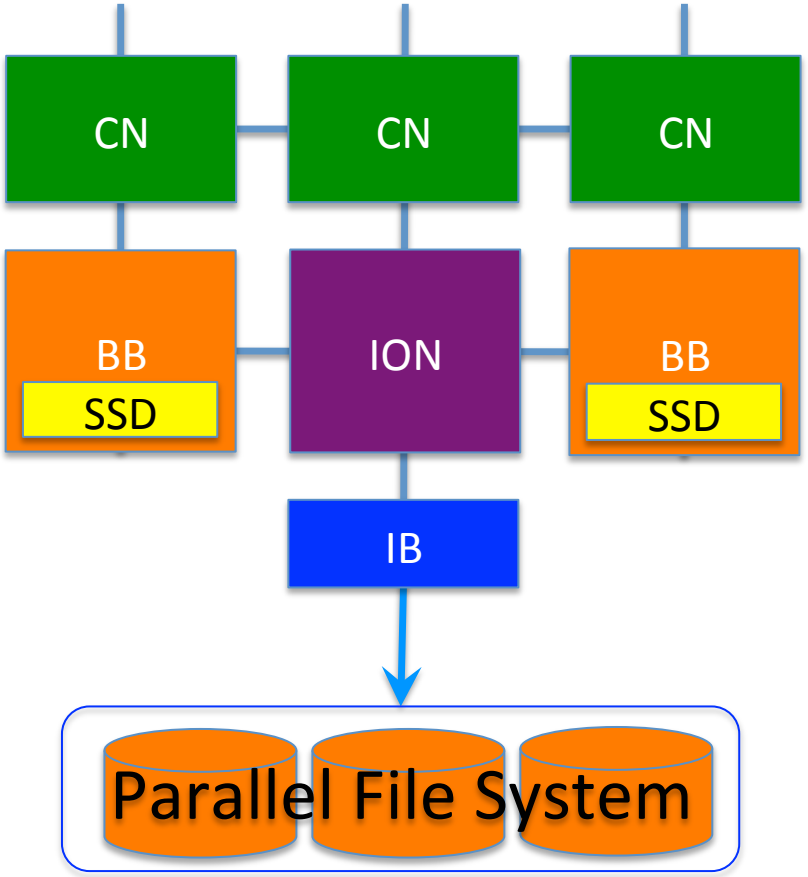




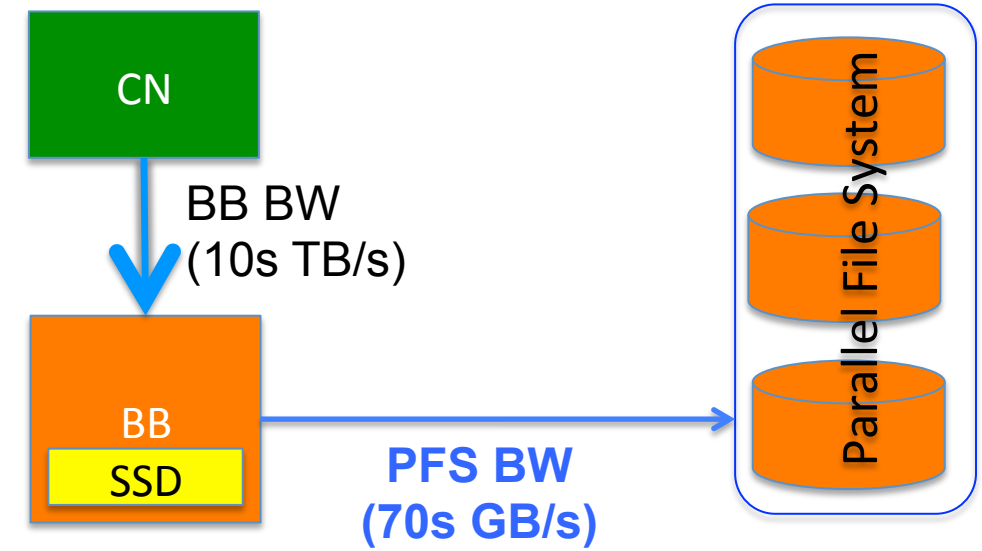


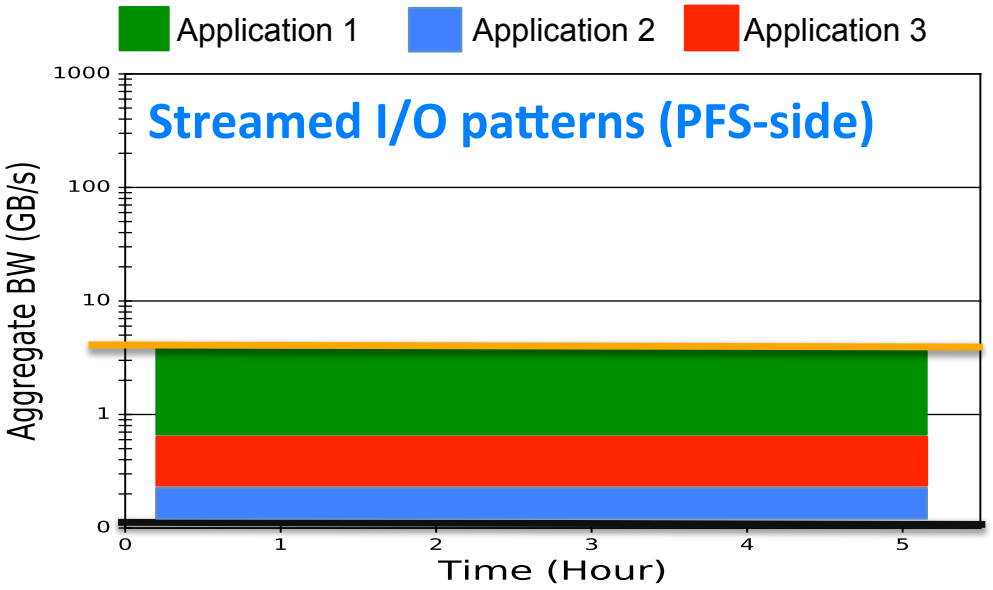
Based on: Liu, N, Cope, J, Carns, P, Carothers, C, Ross, R, Grider, G, Crume, A, Maltzahn, C . "On the Role of Burst Buffers in Leadership-class Storage Systems" MSST/SNAPI 2012

Burst Buffer System



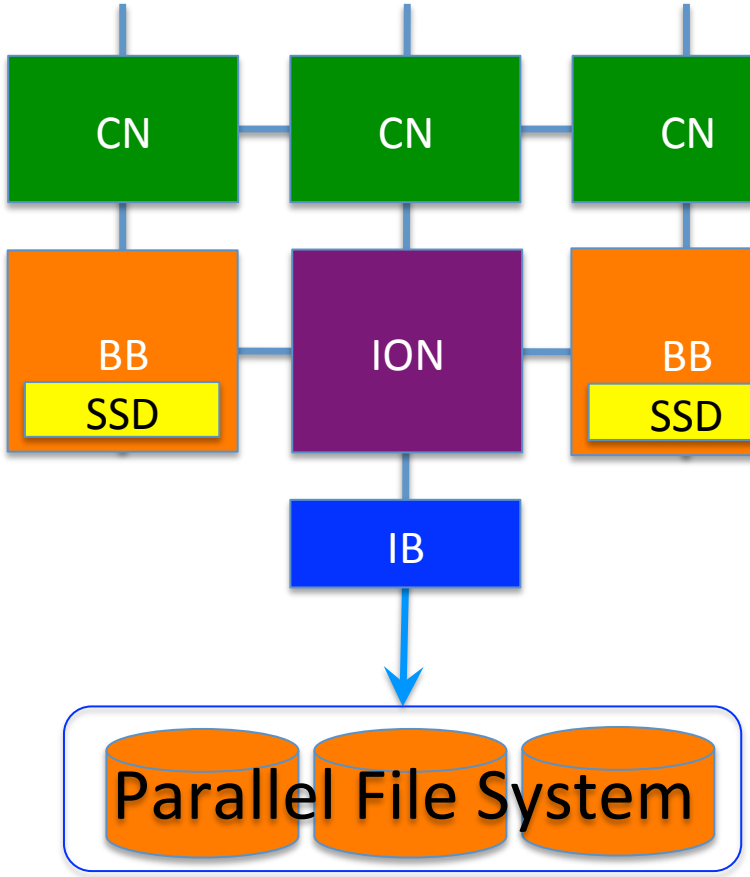
Based on: *http://www.nersc.gov/users/computational-systems/cori/burst-buffer/burst-buffer/*



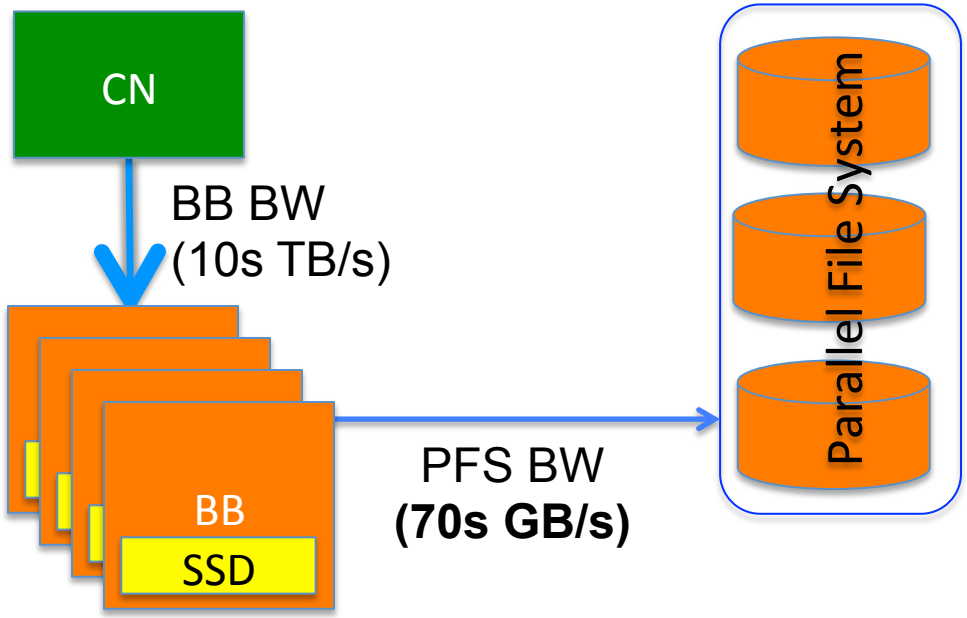


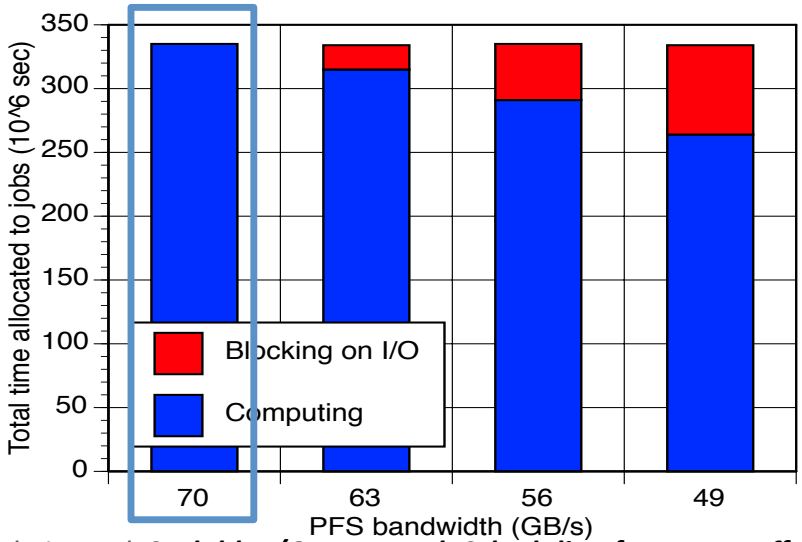
Based on: Liu, N, Cope, J, Carns, P, Carothers, C, Ross, R, Grider, G, Crume, A, Maltzahn, C . "On the Role of Burst Buffers in Leadership-class Storage Systems" MSST/SNAPI 2012

PFS Bottleneck



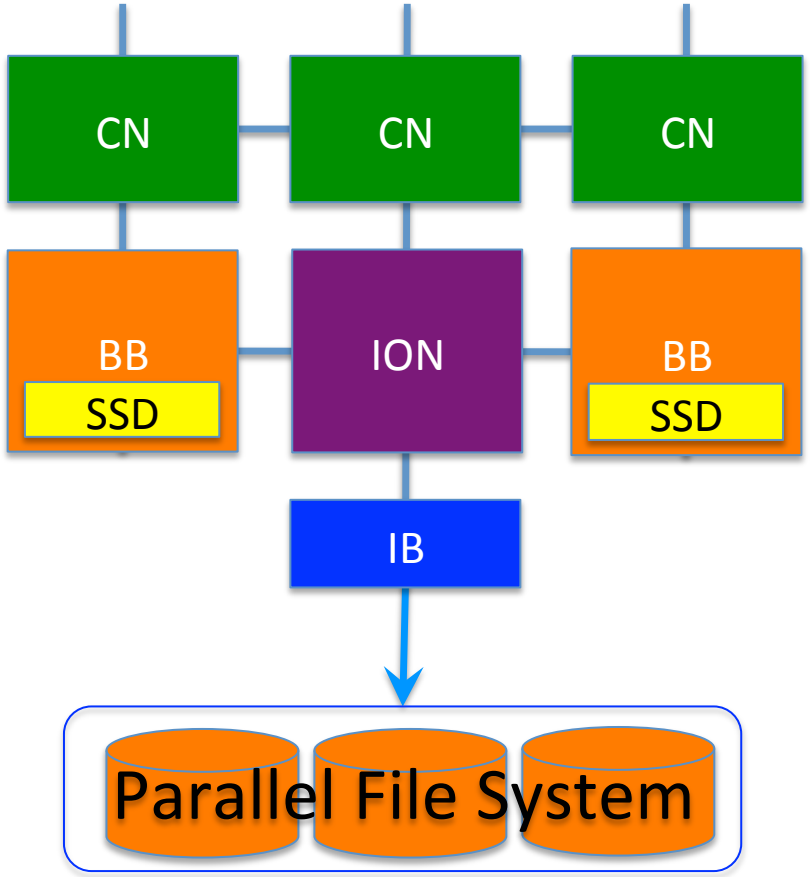
Based on: *http://www.nersc.gov/users/computational-systems/cori/burst-buffer/burst-buffer/*



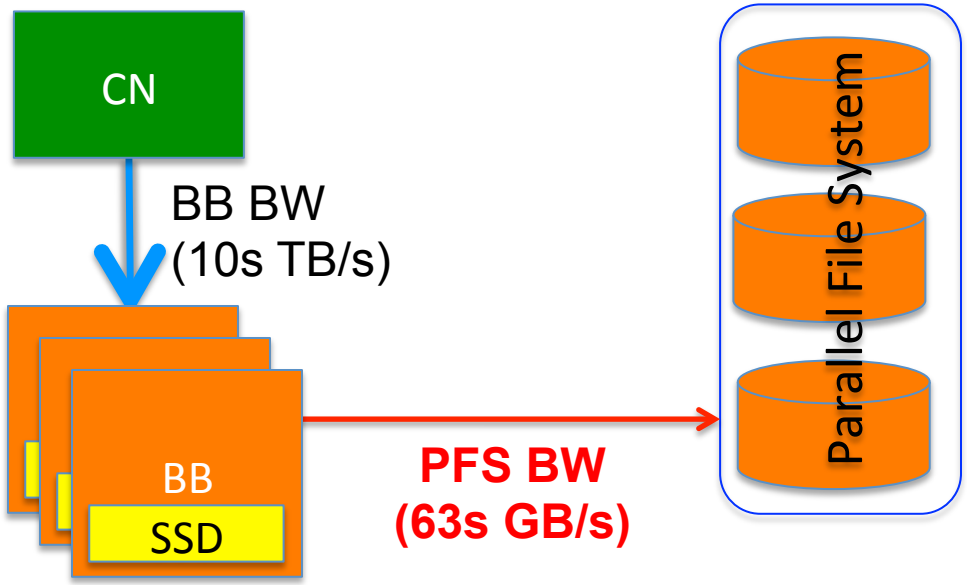


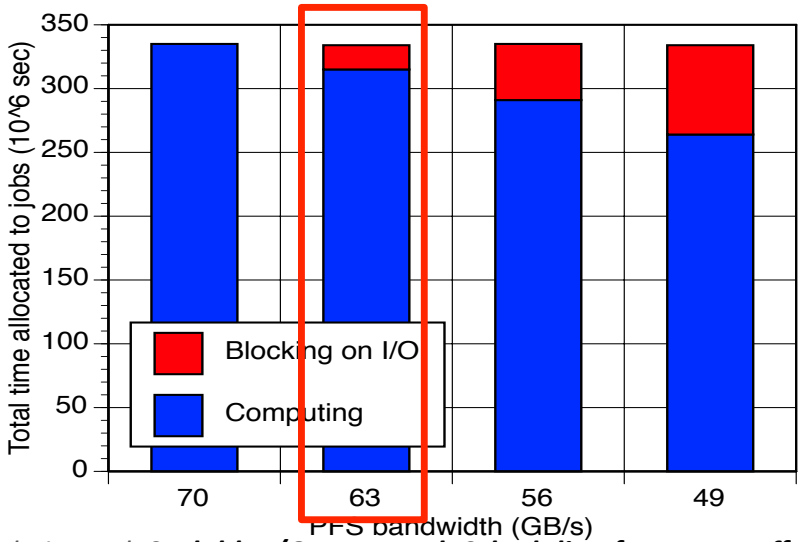
Herbein et al. ***Scalable I/O-aware Job Scheduling for Burst Buffer Enabled HPC Clusters****, HPDC 2016.*

PFS Bottleneck



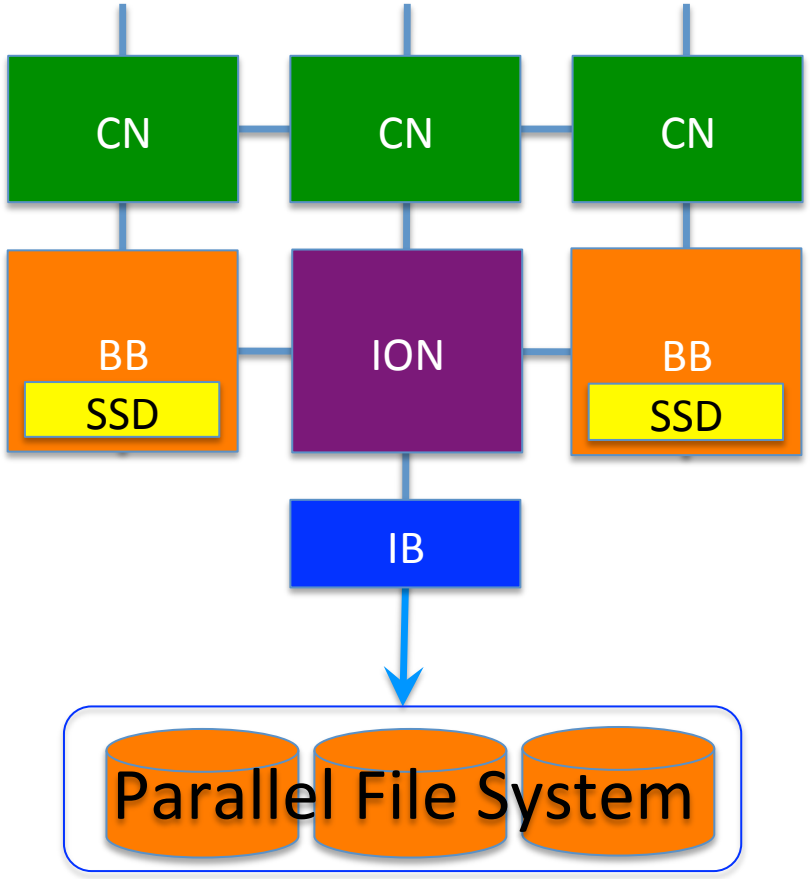
Based on: *http://www.nersc.gov/users/computational-systems/cori/burst-buffer/burst-buffer/*



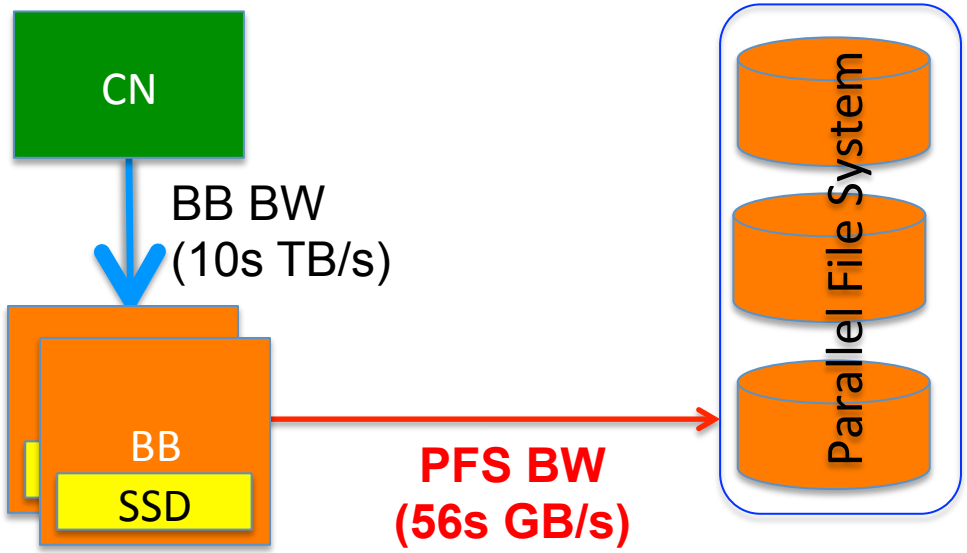


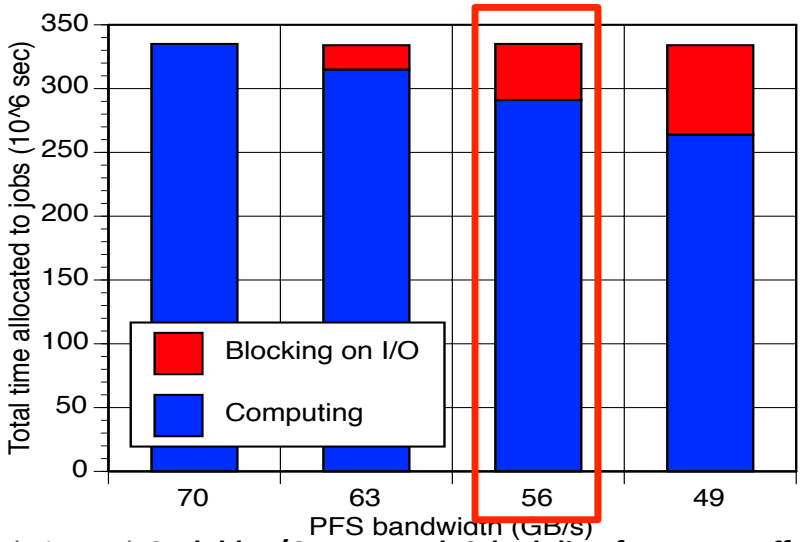
Herbein et al. ***Scalable I/O-aware Job Scheduling for Burst Buffer Enabled HPC Clusters****, HPDC 2016.*

PFS Bottleneck



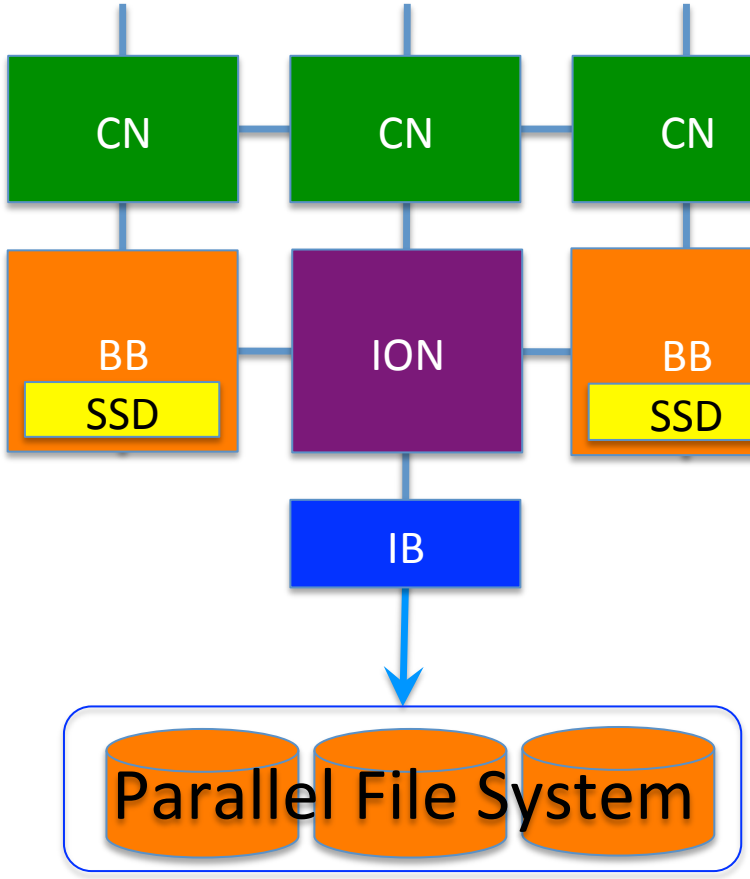
Based on: *http://www.nersc.gov/users/computational-systems/cori/burst-buffer/burst-buffer/*



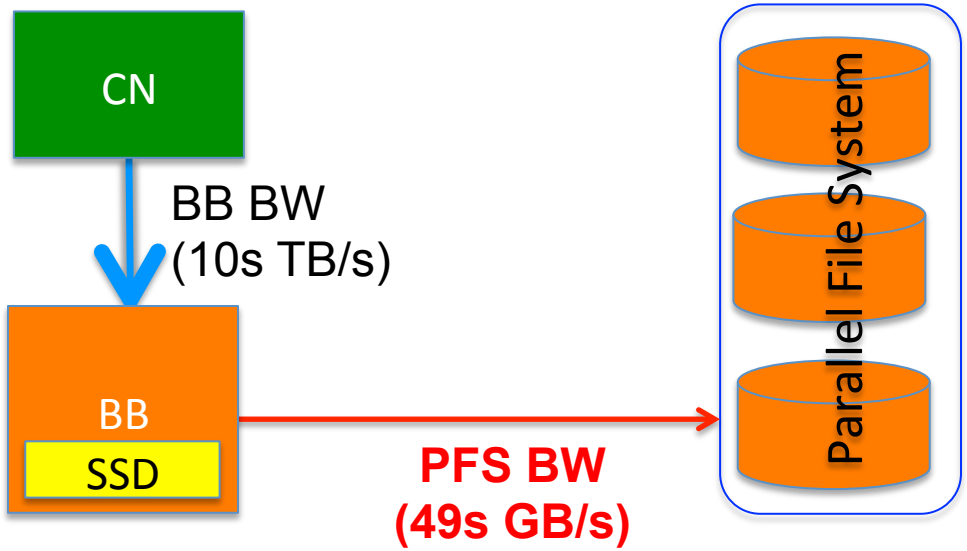


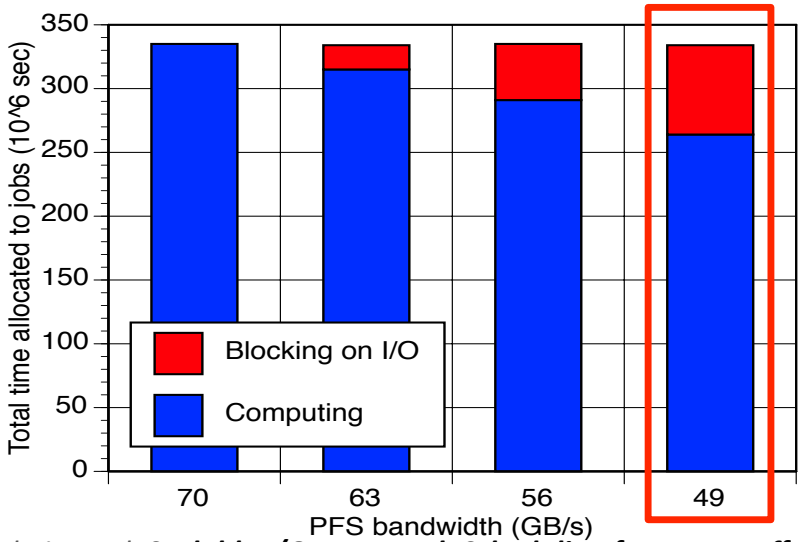
Herbein et al. ***Scalable I/O-aware Job Scheduling for Burst Buffer Enabled HPC Clusters****, HPDC 2016.*

PFS Bottleneck



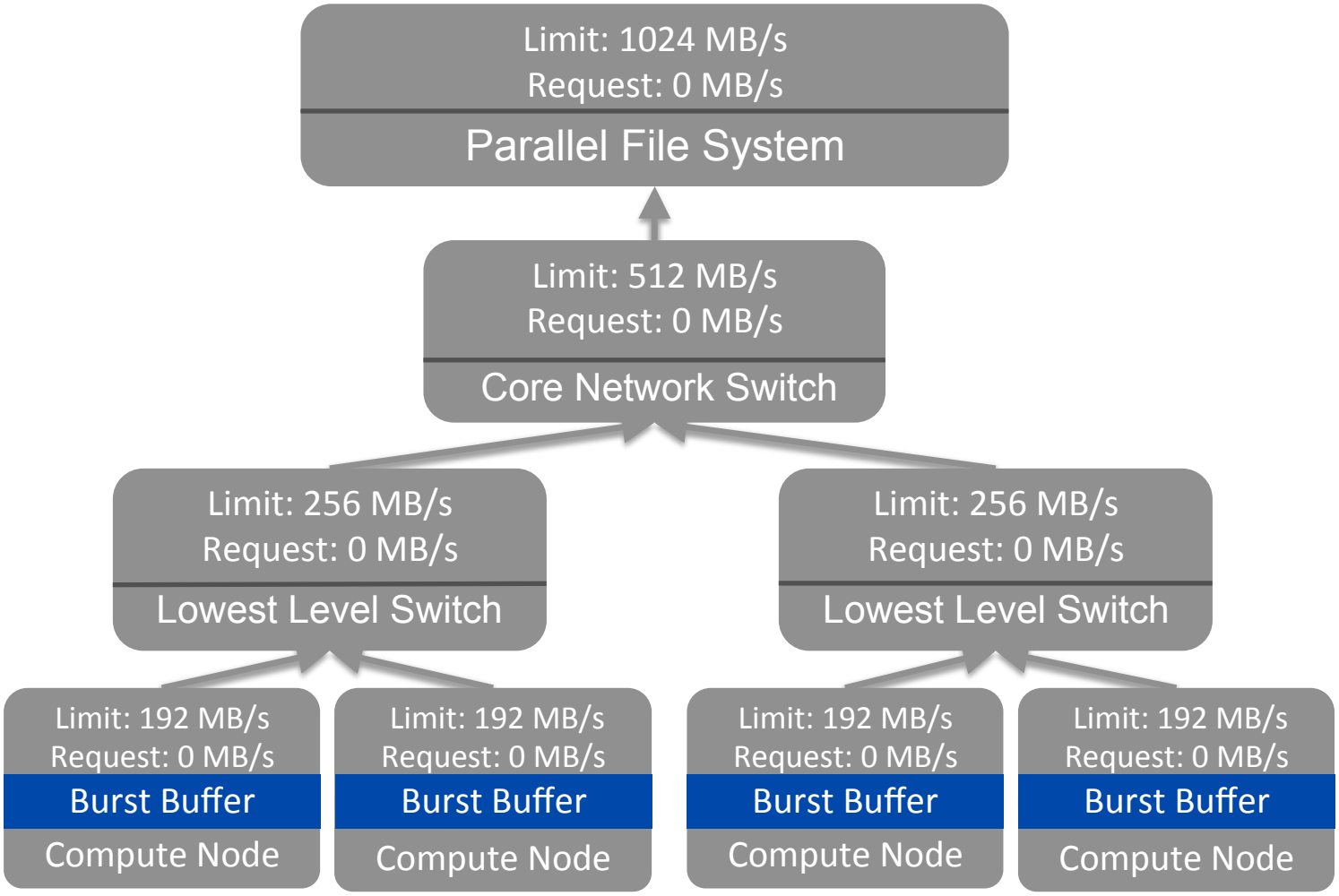
Based on: *http://www.nersc.gov/users/computational-systems/cori/burst-buffer/burst-buffer/*



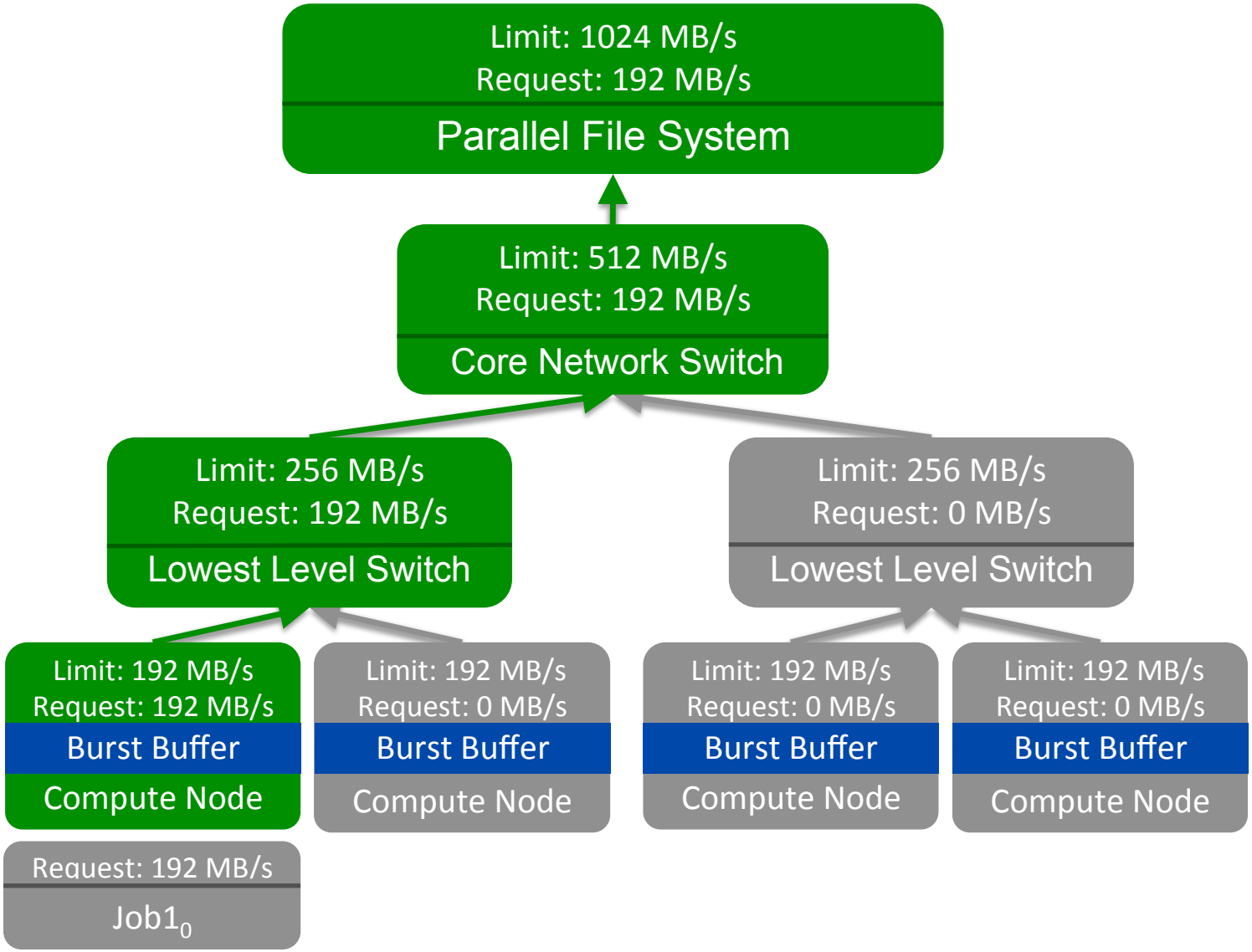


Herbein et al. ***Scalable I/O-aware Job Scheduling for Burst Buffer Enabled HPC Clusters****, HPDC 2016.*

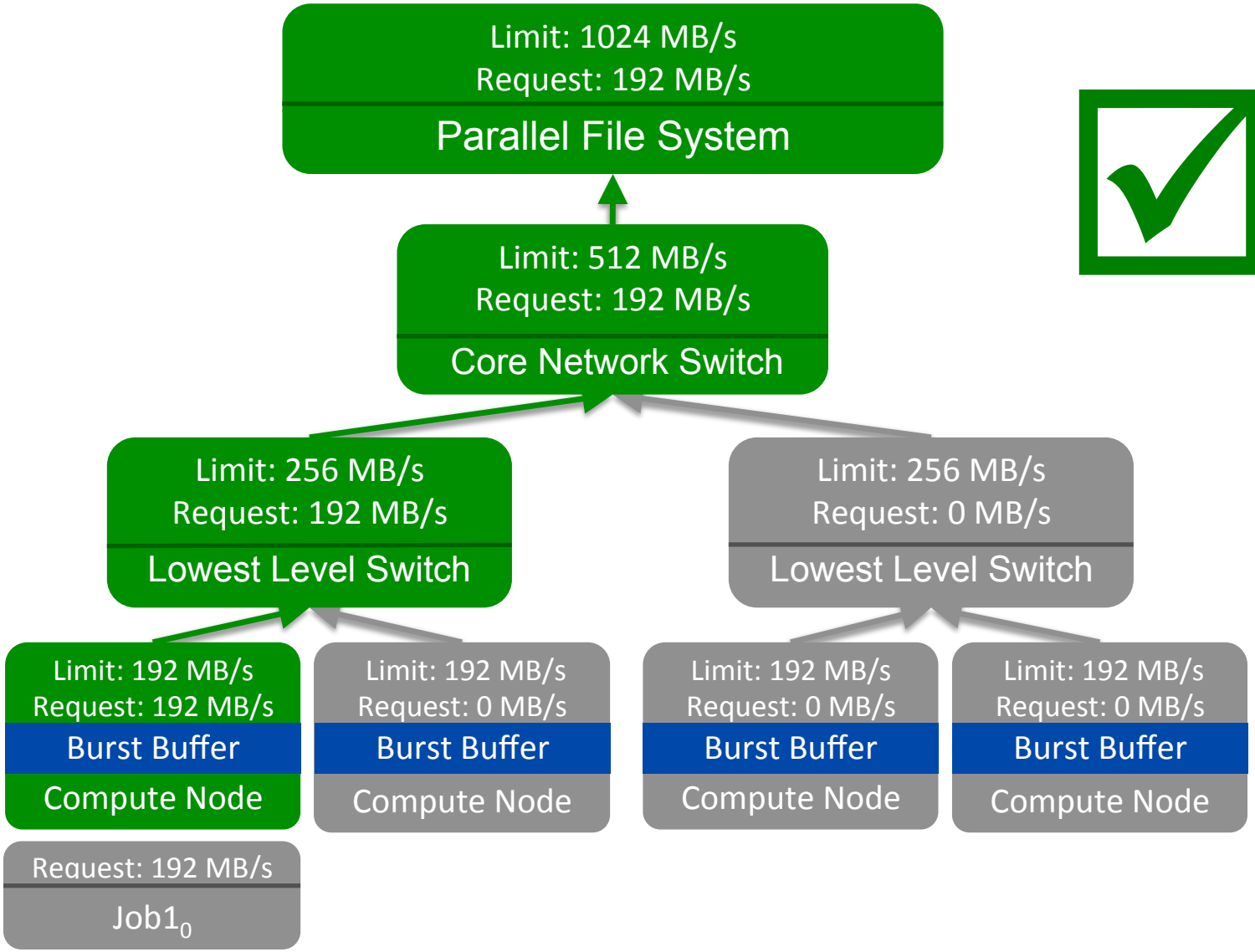
Integrate I/O-awareness in Flux Scheduler



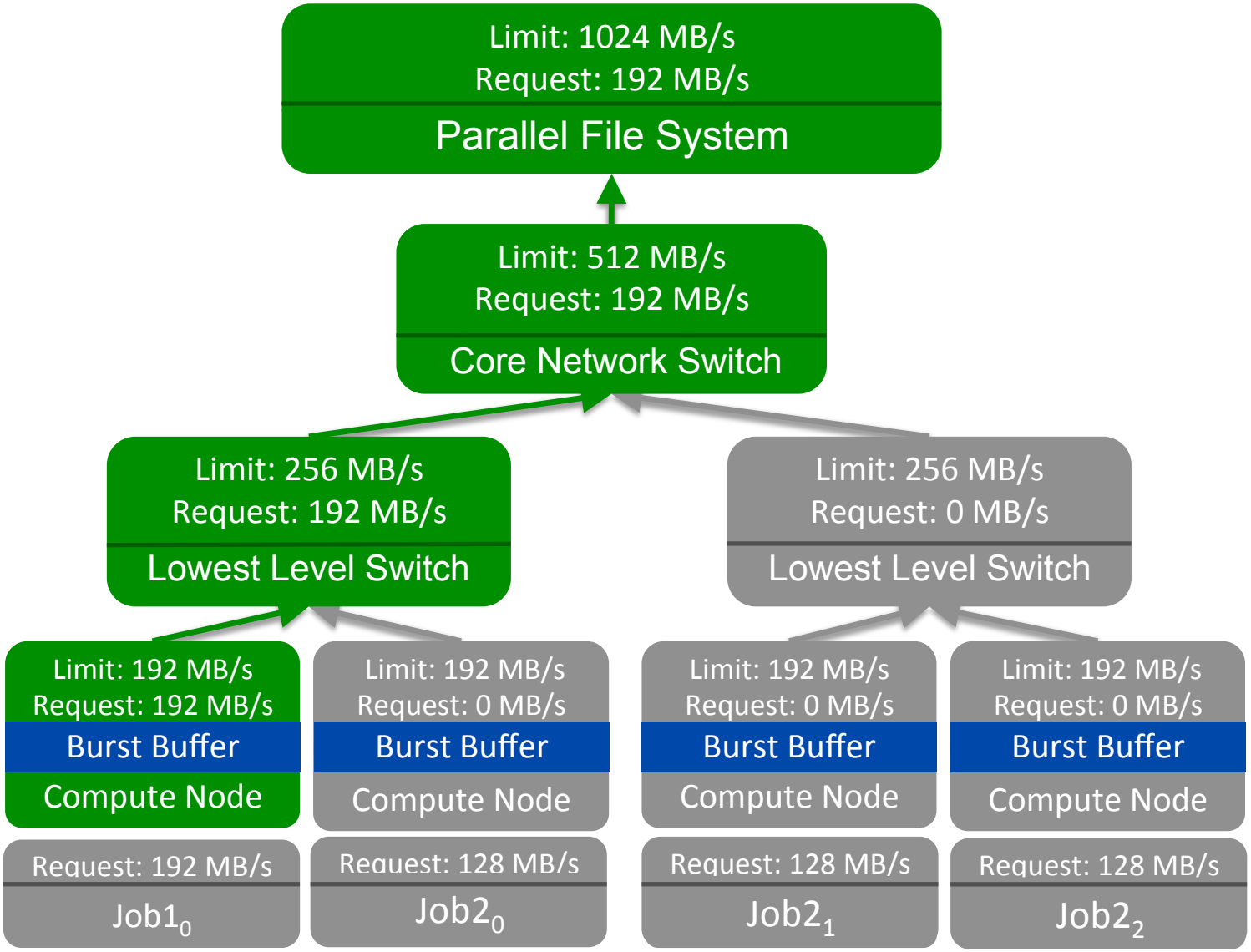
Integrate I/O-awareness in Flux Scheduler



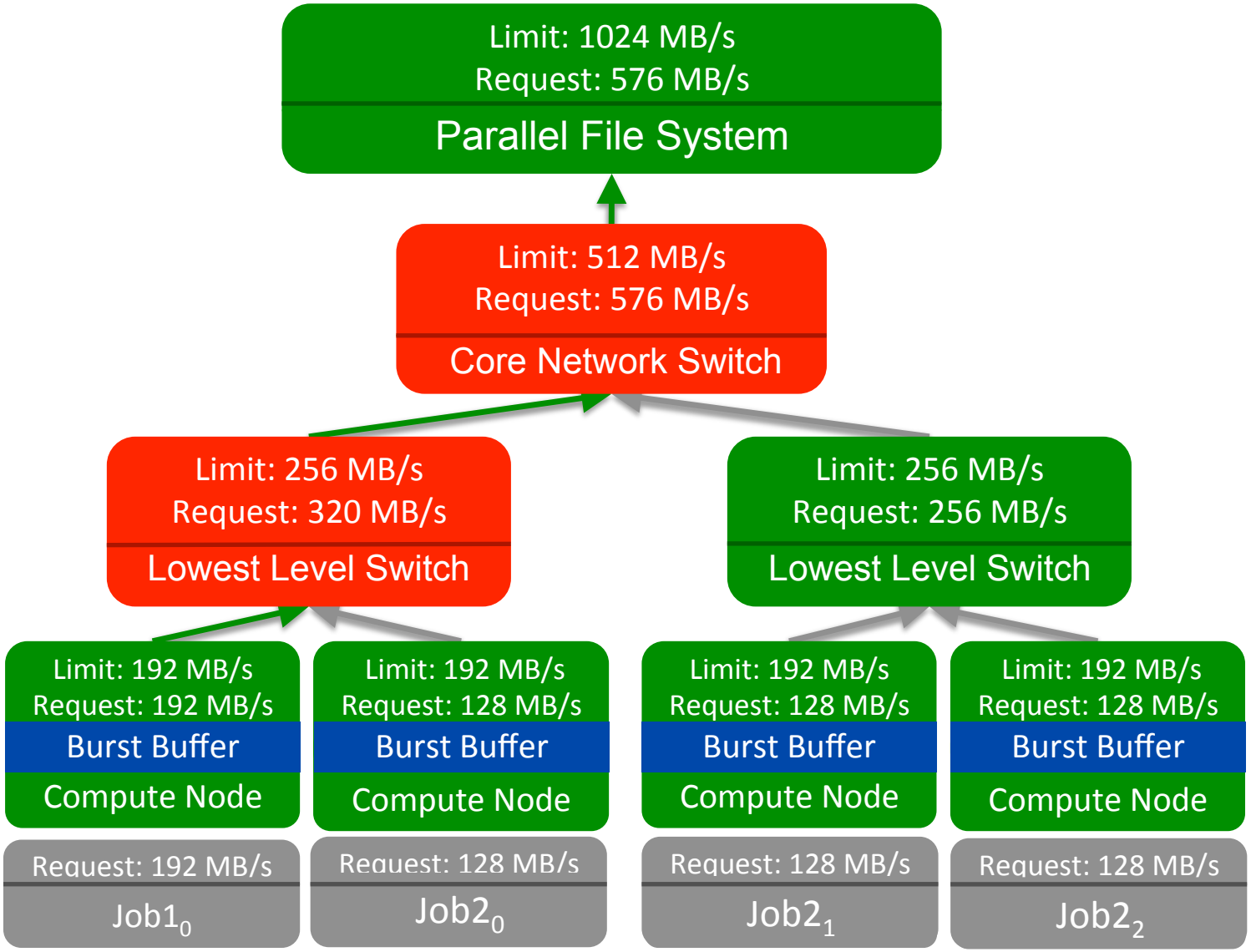
Integrate I/O-awareness in Flux Scheduler



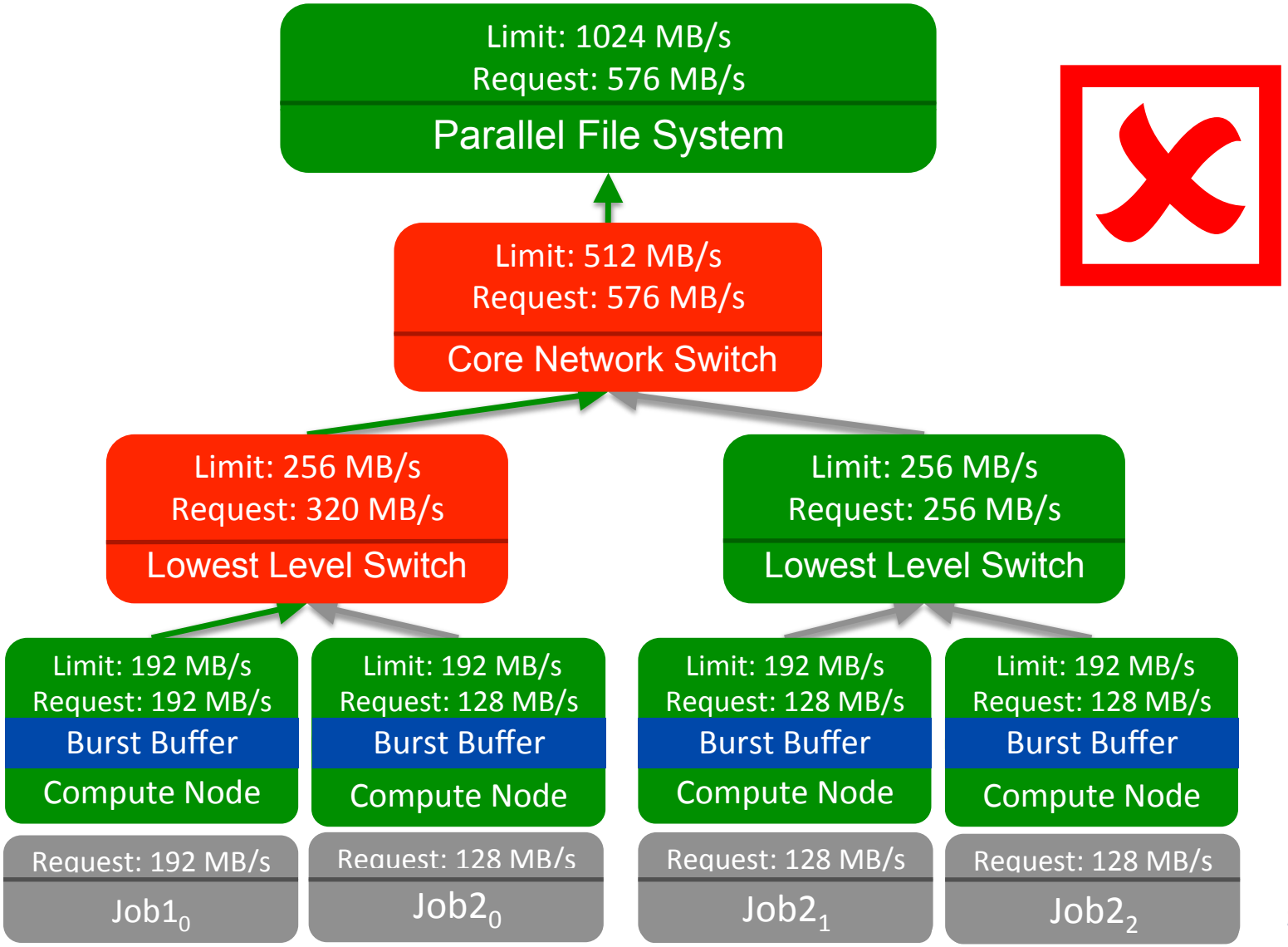
Integrate I/O-awareness in Flux Scheduler



Integrate I/O-awareness in Flux Scheduler

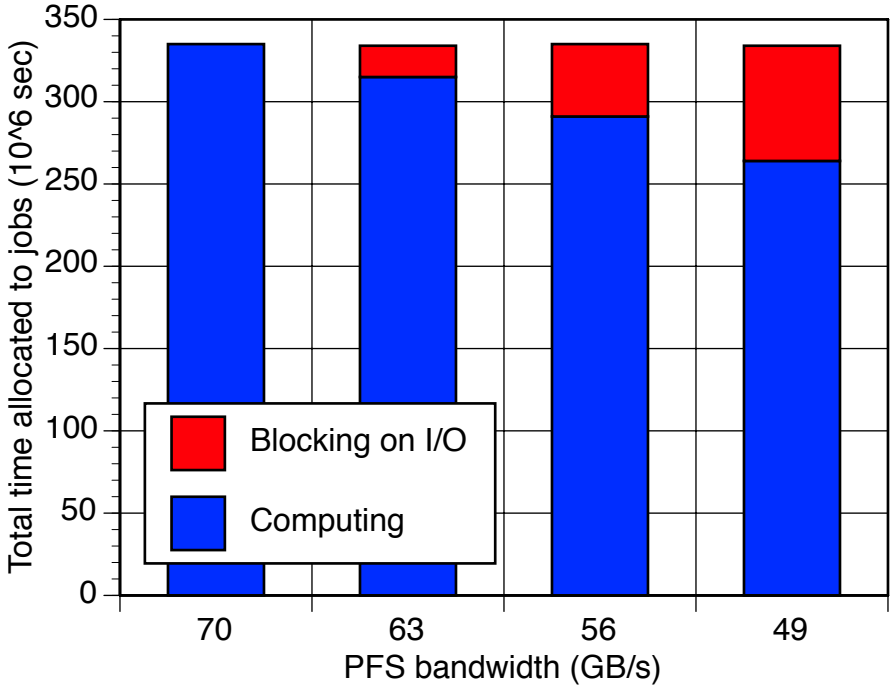


Integrate I/O-awareness in Flux Scheduler

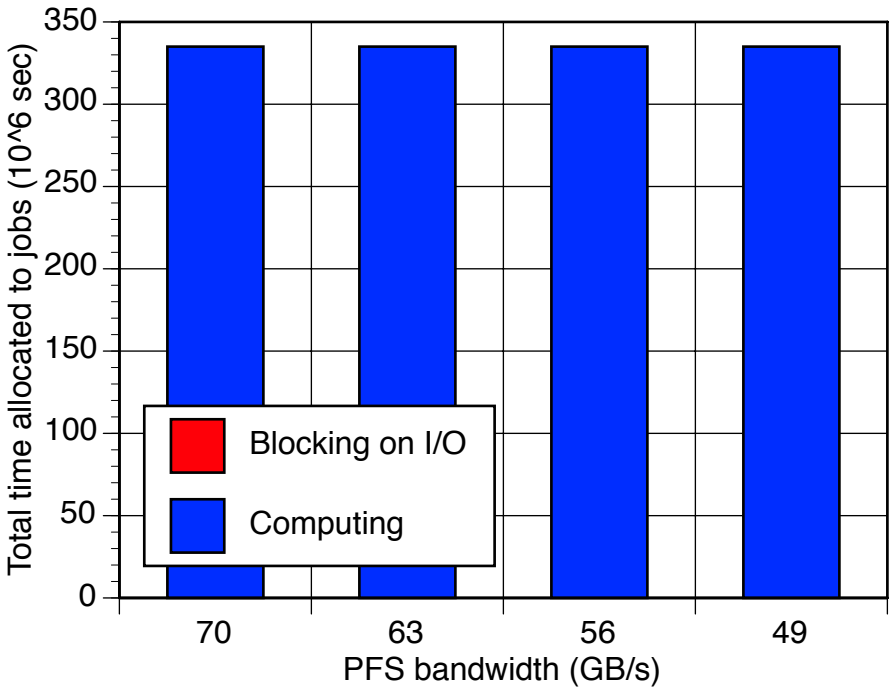


I/O-Ignorant vs. I/O-Aware Scheduling in Flux

I/O-ignorant scheduler



I/O-aware scheduler



I/O-Aware scheduling results in 100% of application time to be spent in computation

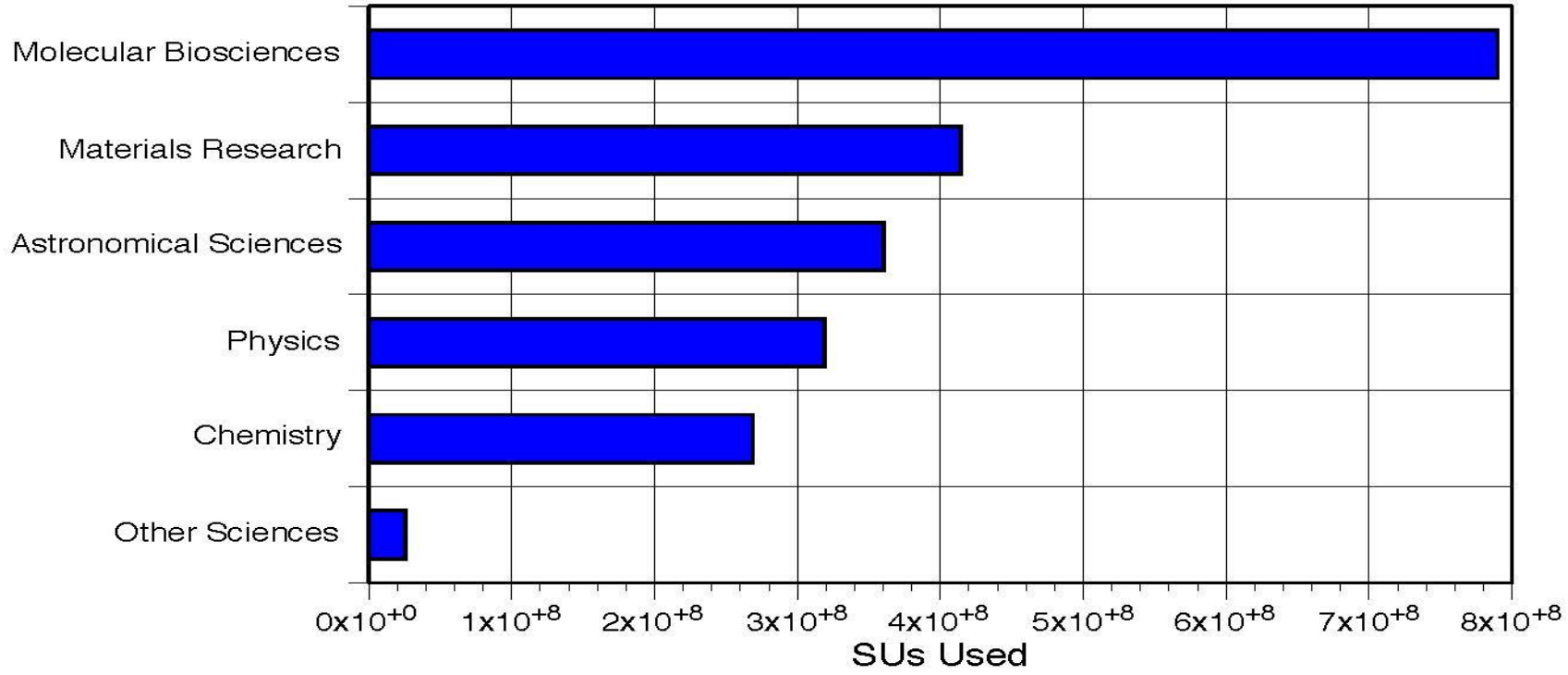
Herbein et al. ***Scalable I/O-aware Job Scheduling for Burst Buffer Enabled HPC Clusters****, HPDC 2016.*

Challenges

- Burst Buffers are not the magic I/O silver bullet
  - I/O contention still a problem if we exceed the burst buffer capability
  - Burst buffers improve offloading bandwidth but do **NOT** help uploading data from storage for runtime analysis

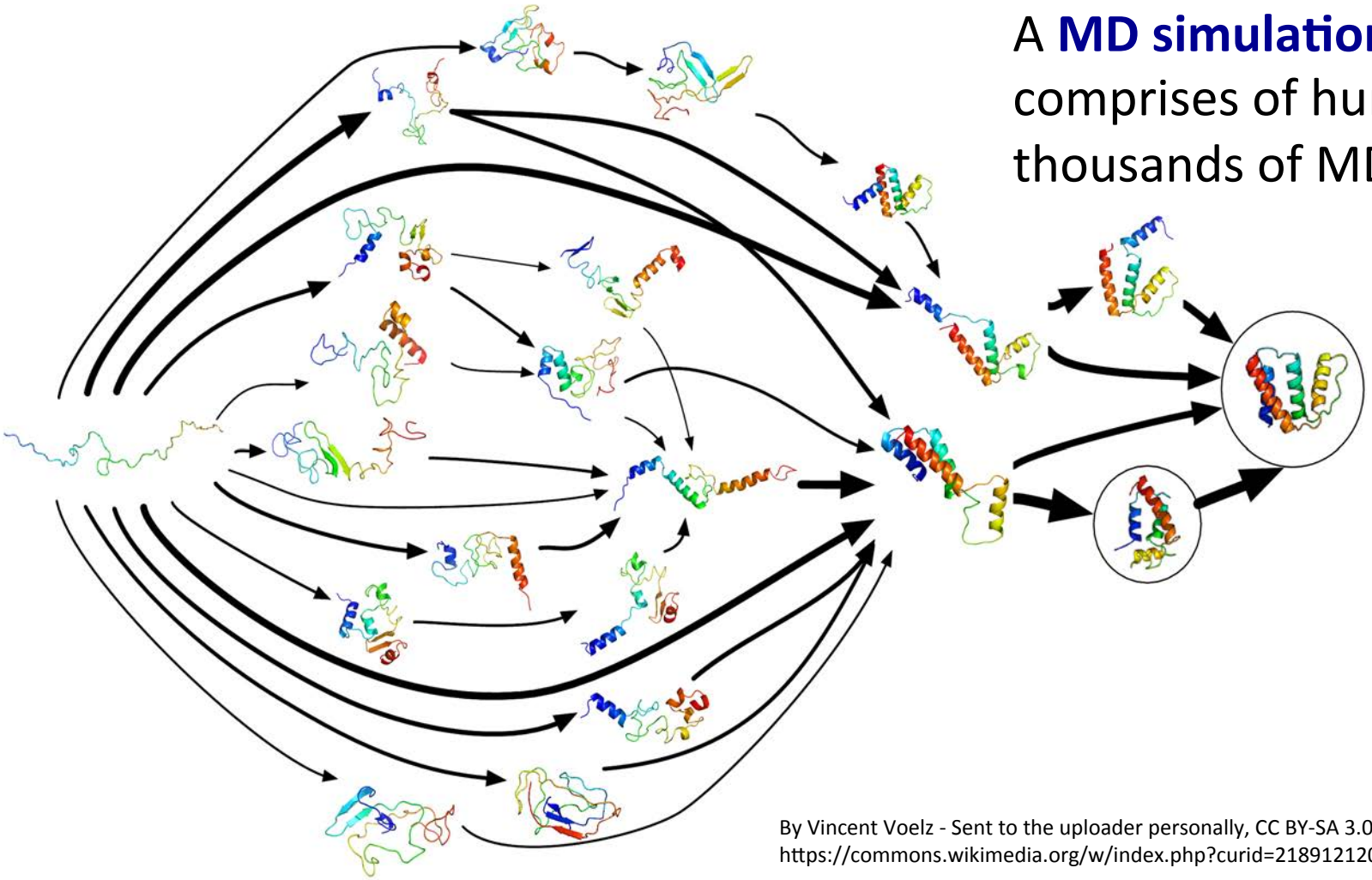
MD Simulations are Alive and Kicking!

XSEDE SUs used by type of targeted science over the past 6 months (March 1, 2016 - August 31, 2016)



Four of the top 10 XSEDE users run molecular simulations (i.e., Schulten at UIUC, Feig at Michigan State U, Voth at U Chicago, and Case at Rutgers U)

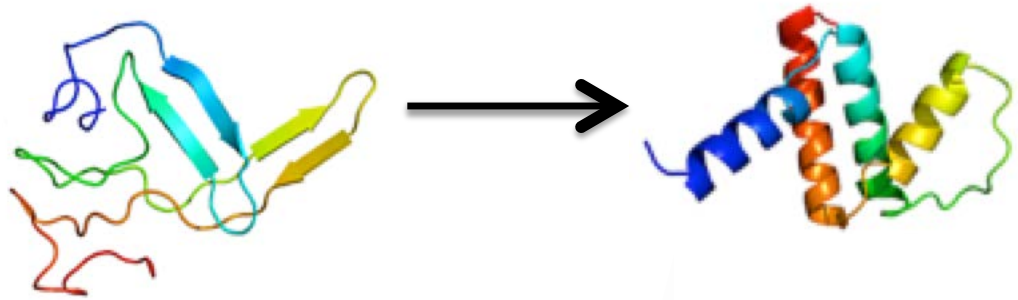
MD Simulations as an Ensemble of HPC Jobs



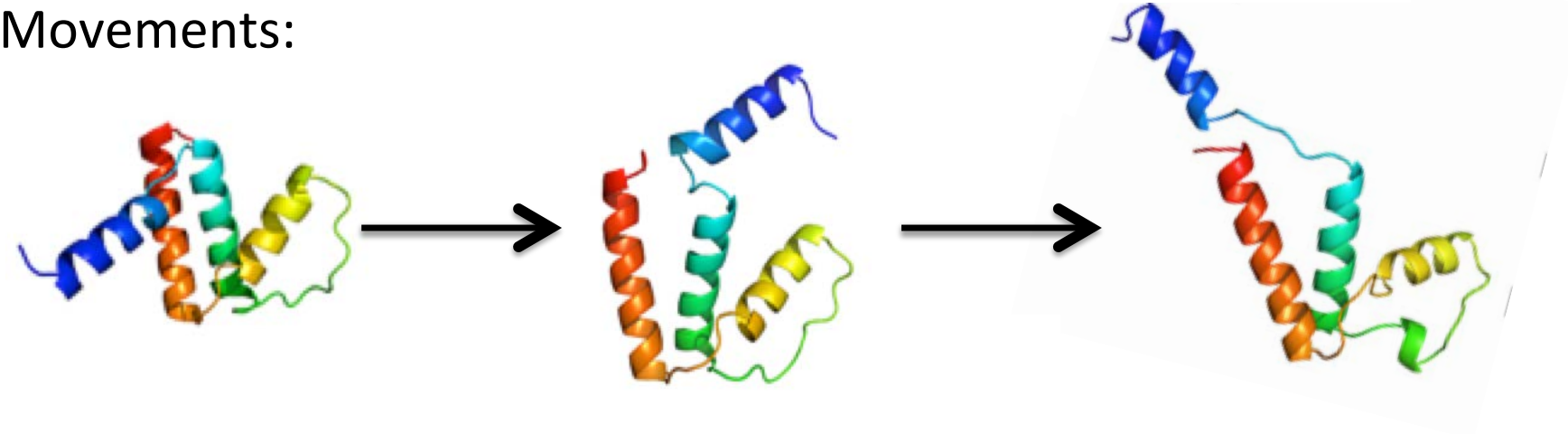
A **MD simulation** comprises of hundreds of thousands of MD job

Capturing Rare Events

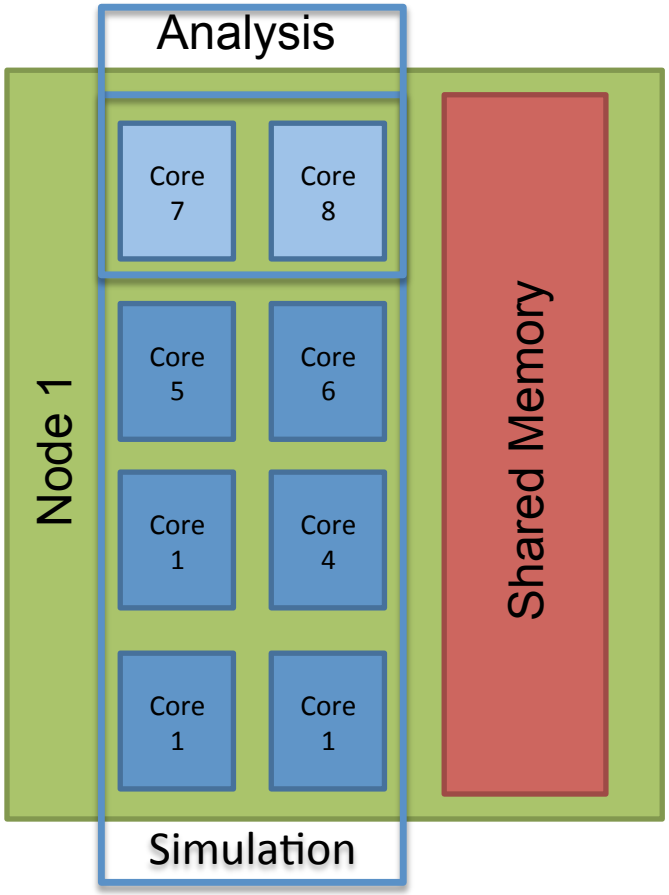
Transformations:

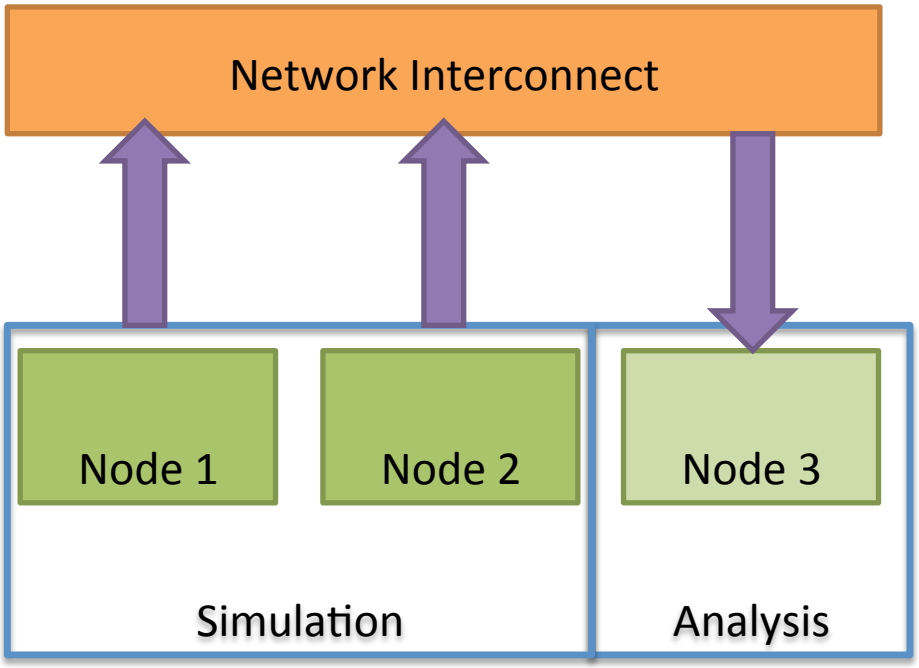


Movements:



In-situ and *In-transit* Analysis



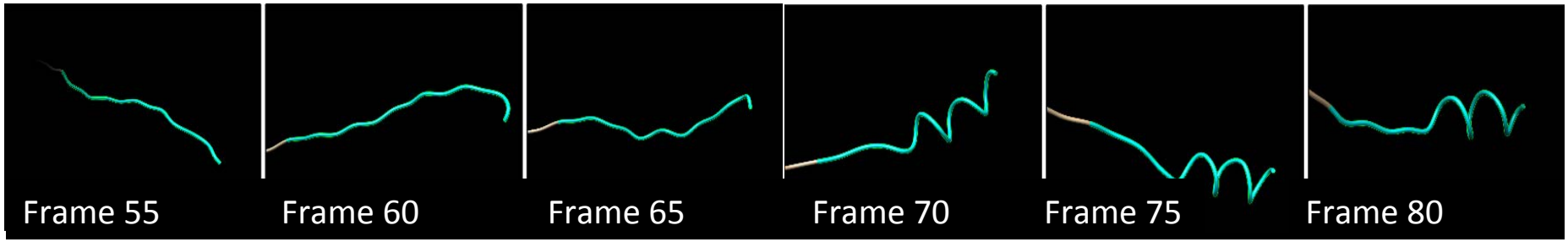


Example of tools:

- DataSpaces (Rutgers U.)
- DataStager (GeorgiaTech)

Requirements to Capture Rare Events *In Situ*

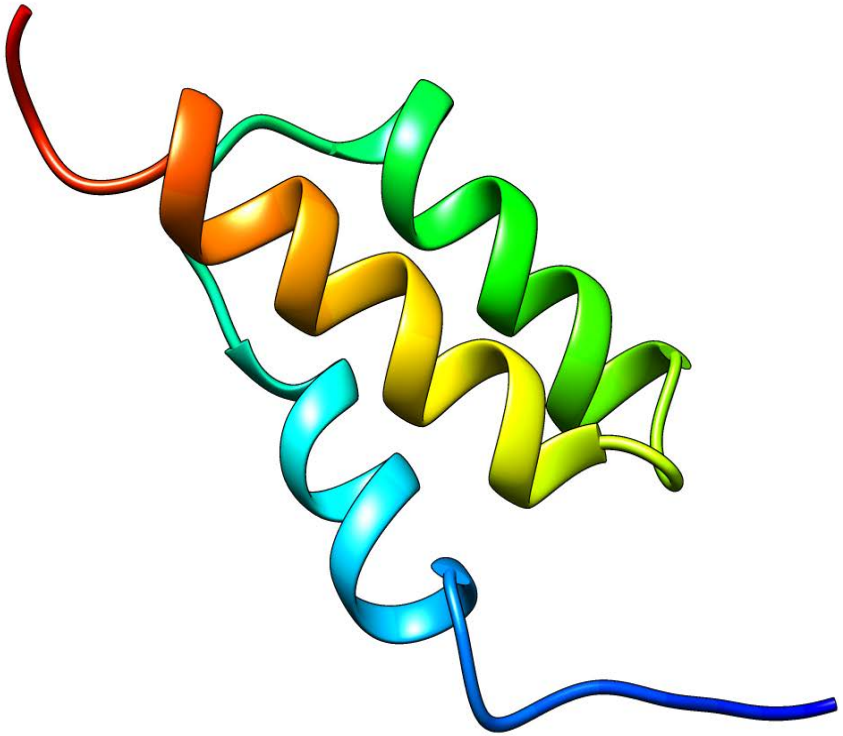
Frames (or snapshots) of an MD trajectory:



- We want to capture what is going on in each frame **without**:
  - Disrupting the simulation (e.g., stealing CPU and memory on the node)
  - Moving all the frames to a central file system and analyzing them once the simulation is over
  - Comparing each frame with past frames of the same job
  - Comparing each frame with frames of other jobs

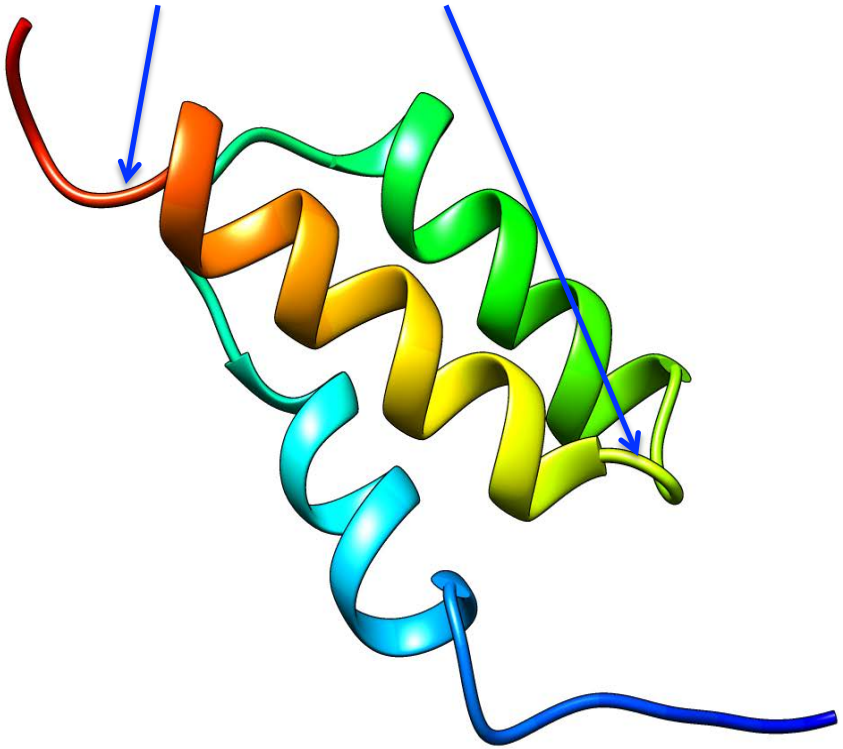
Capturing the Transformations in a Structure

Given a frame of an
MD job at time *t*



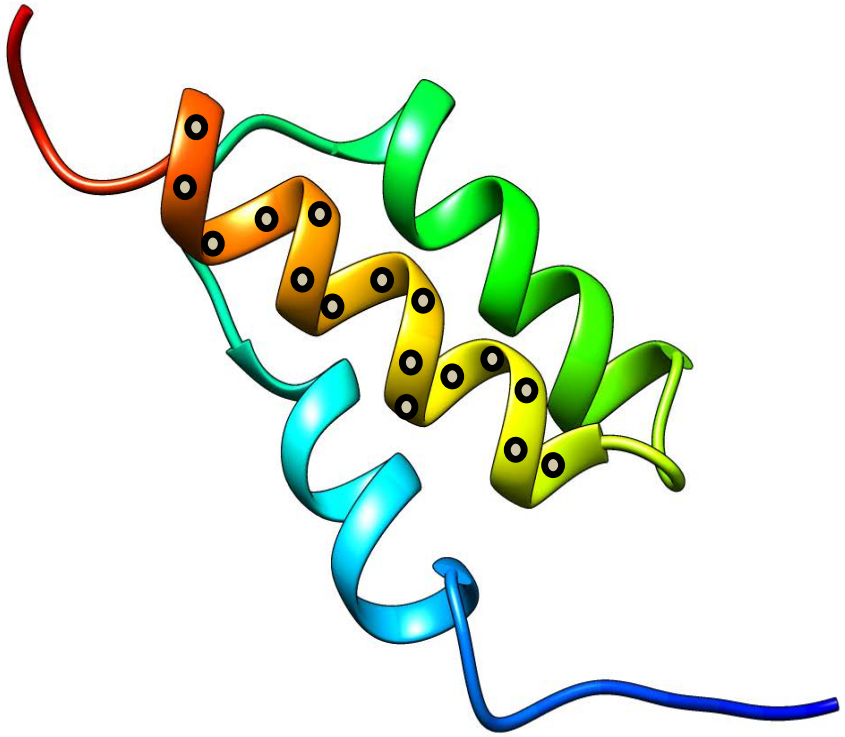
Capturing the Transformations in a Structure

Define the substructure:
start and **stop** amino acids



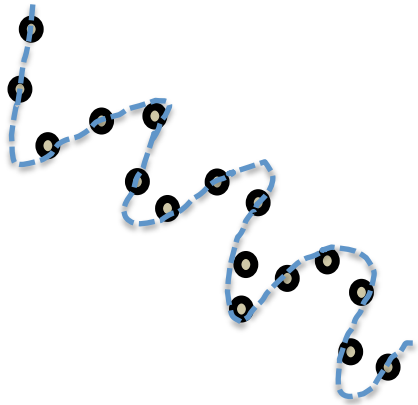
Capturing the Transformations in a Structure

Drop all but not the backbone
atoms of the structure (C^{α} atoms)



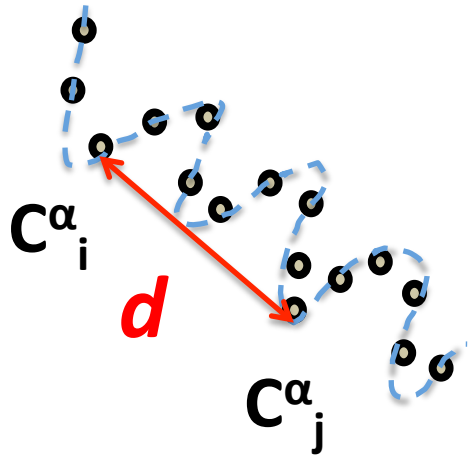
Capturing the Transformations in a Structure

Drop all but not the backbone
atoms of the structure (C^{α} atoms)



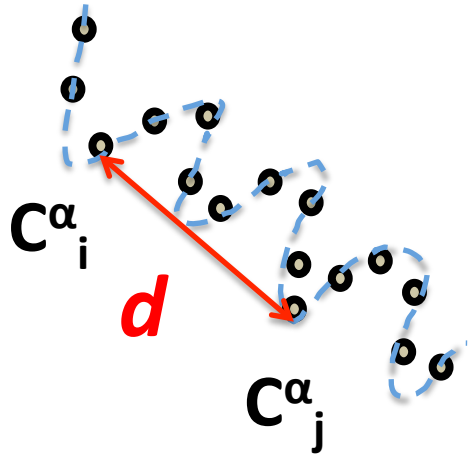
Capturing the Transformations in a Structure

Measure the distance between C^{α}_{j} and C^{α}_{i}



Capturing the Transformations in a Structure

Measure the distance between C^{α}_{j} and C^{α}_{i}



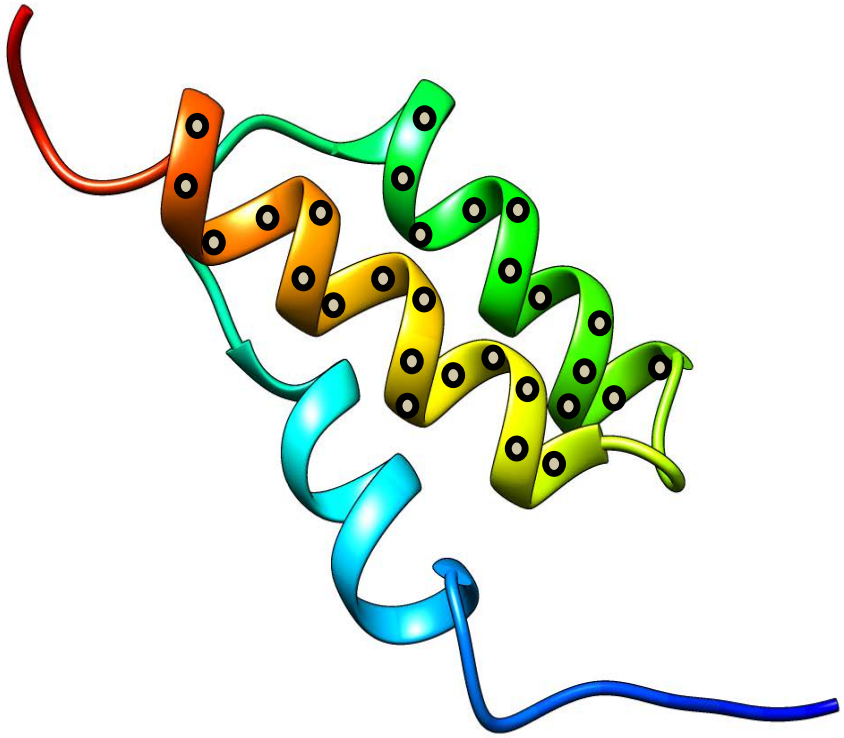
Build the **substructure Euclidean Distance Matrix (D)**

$$D = \begin{array}{c} \\ C^{\alpha}_{j} \\ \\ \\ \\ \\ \end{array} \begin{array}{c} C^{\alpha}_{i} \\ \begin{bmatrix} 0 & \times & \times & \times & \times & \times \\ \times & 0 & d & \times & \times & \times \\ \times & d & 0 & \times & \times & \times \\ \times & \times & \times & 0 & \times & \times \\ \times & \times & \times & \times & 0 & \times \\ \times & \times & \times & \times & \times & 0 \end{bmatrix} \end{array}$$

Compute largest eigenvalue → λ_{max}

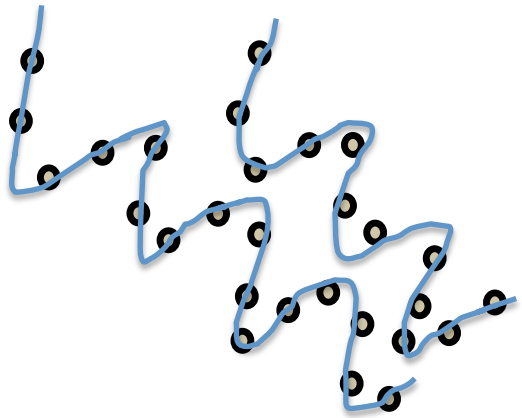
Capturing Movements between Structures

Drop all but not the backbone atoms of the two structure



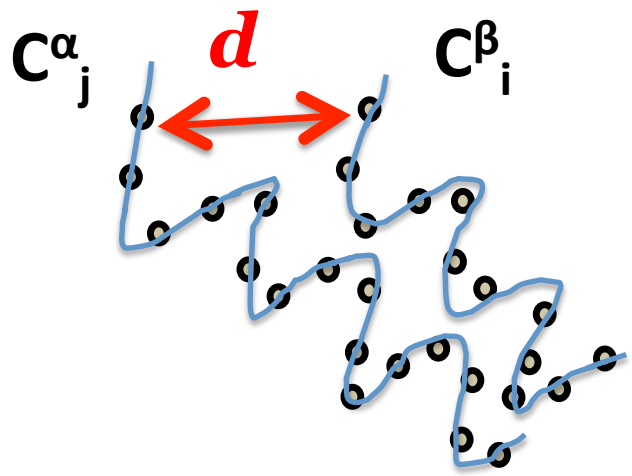
Capturing Movements between Structures

Drop all but not the backbone atoms of the two structure



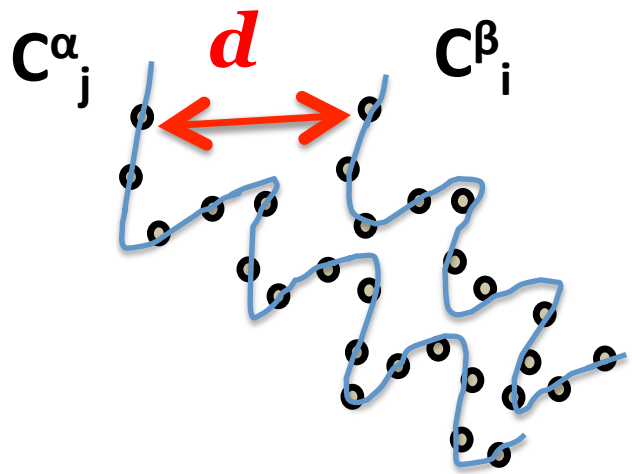
Capturing Movements between Structures

Measure the distance between C^{α}_{j} and C^{β}_{i}



Capturing Movements between Structures

Measure the distance between C^{α}_{j} and C^{β}_{i}



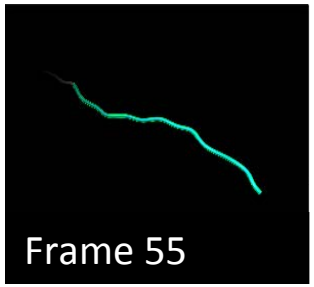
Build a **bipartite distance matrix** by comparing two substructures

$$D = \begin{array}{c} \\ j \end{array} \overset{i}{\begin{bmatrix} 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & d & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ \times & d & \times & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 \end{bmatrix}}$$

Compute largest eigenvalue → λ_{max}

Proxy for Rare Events

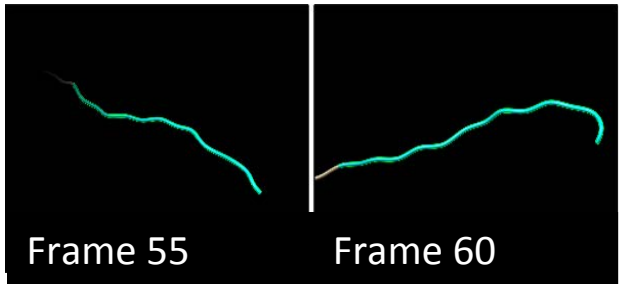
Frames of an MD job:



λ_{55}

Proxy for Rare Events

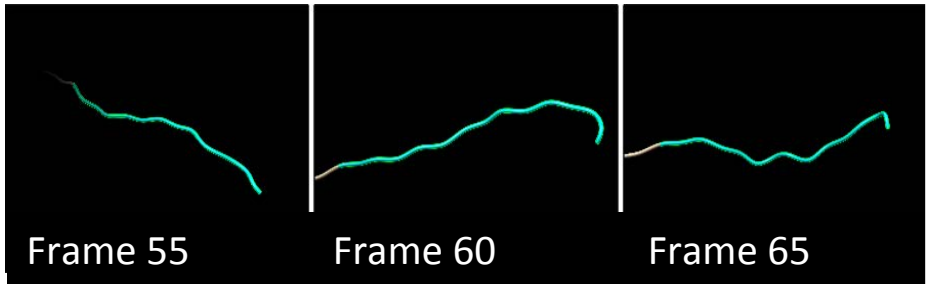
Frames of an MD job:



λ_{55} λ_{60}

Proxy for Rare Events

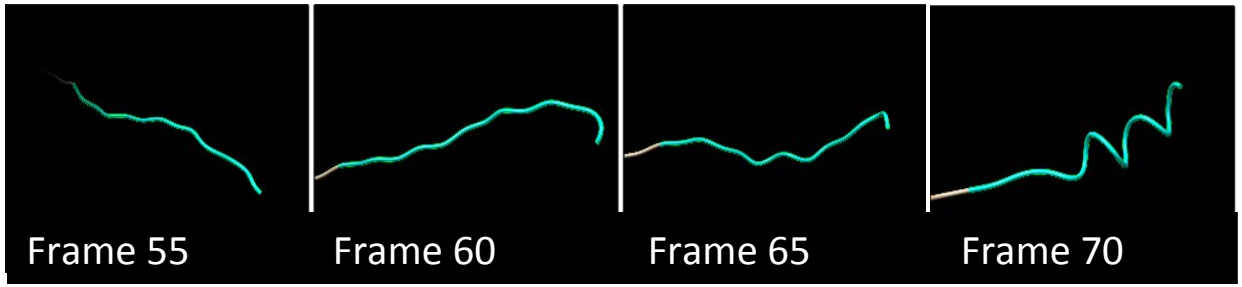
Frames of an MD job:



λ_{55} λ_{60} λ_{65}

Proxy for Rare Events

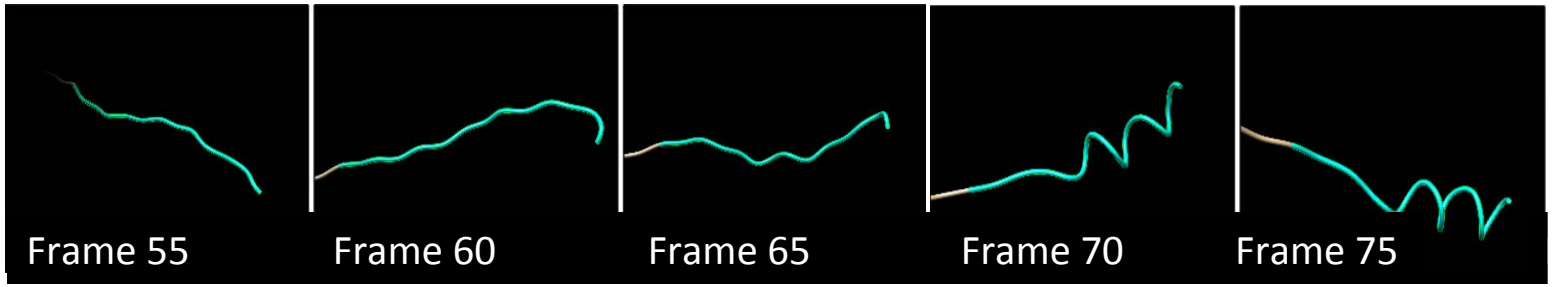
Frames of an MD job:



λ_{55} λ_{60} λ_{65} λ_{70}

Proxy for Rare Events

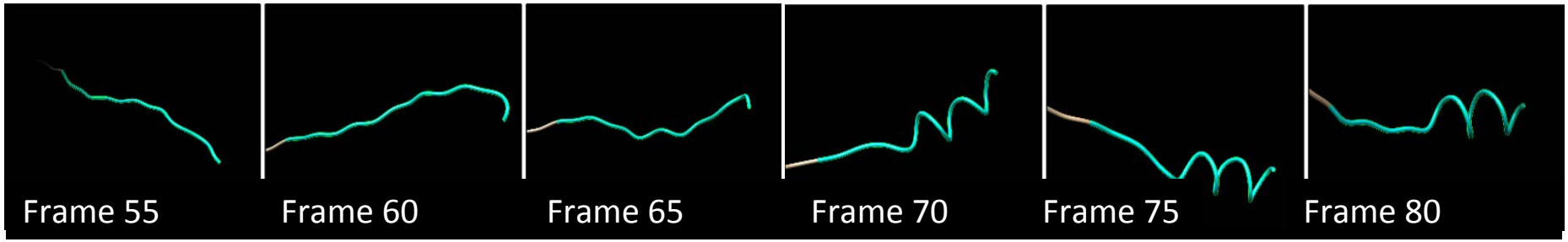
Frames of an MD job:



λ_{55} λ_{60} λ_{65} λ_{70} λ_{75}

Proxy for Rare Events

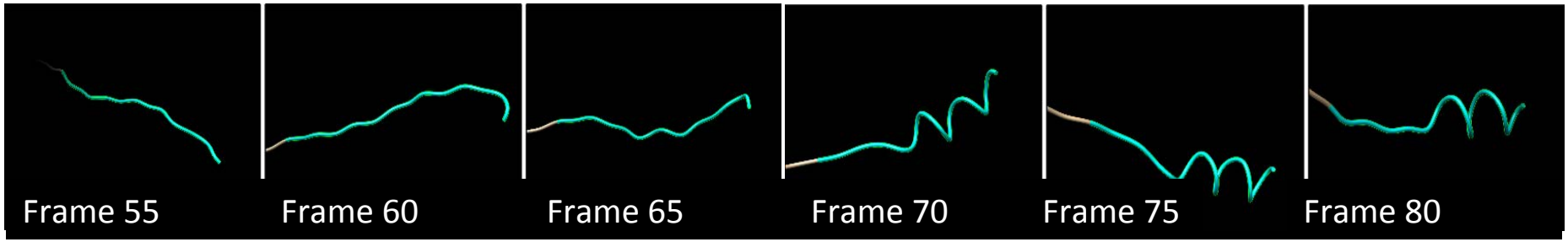
Frames of an MD job:



λ_{55} λ_{60} λ_{65} λ_{70} λ_{75} λ_{85}

Proxy for Rare Events

Frames of an MD job:



λ_{55} λ_{60} λ_{65} λ_{70} λ_{75} λ_{85}

Can the distance between two max eigenvalues serve as a proxy for distance between the two associated conformations?

Reasons to Love Symmetric Matrices

Can the distance between two max eigenvalues serves as a proxy for distance between the two associated conformations?

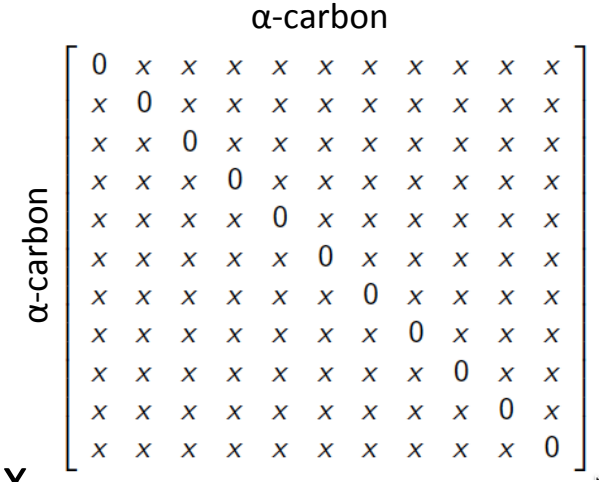
- Euclidean distance matrix D is symmetric
- Eigenvalues of symmetric, real matrices are stable
  - Small perturbations of D result in only small changes in the eigenvalues
  - Euclidean distance matrix is insensitive to rigid transformation
- Use only largest eigenvalue in distance matrix

$\lambda max = \lambda 1 < \lambda 2 < \lambda 3 < \lambda 4 < \lambda 5 = \lambda min$

$\lambda 1 + \lambda 2 + \lambda 3 + \lambda 4 + \lambda 5 = 0$

$\lambda 1 >> \lambda 2 \sim \lambda 3 \sim \lambda 4 \sim 0$

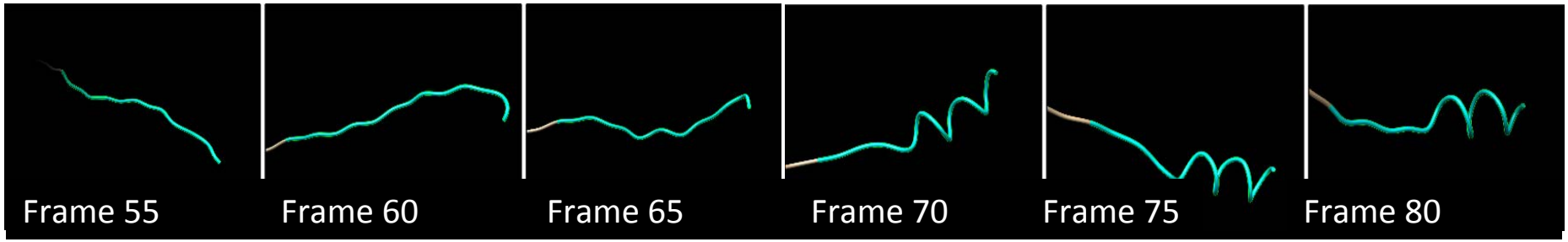
$\lambda max = \lambda 1 \sim -\lambda 5 = -\lambda min$



"***In-Situ Data Analysis and Indexing of Protein Trajectories***," Travis Johnston, Buyu Zhang, Adam Liwo, Silvia Crivelli, and Michela Taufer. JCC 2017.

Proxy for Rare Events

Frames of an MD job:

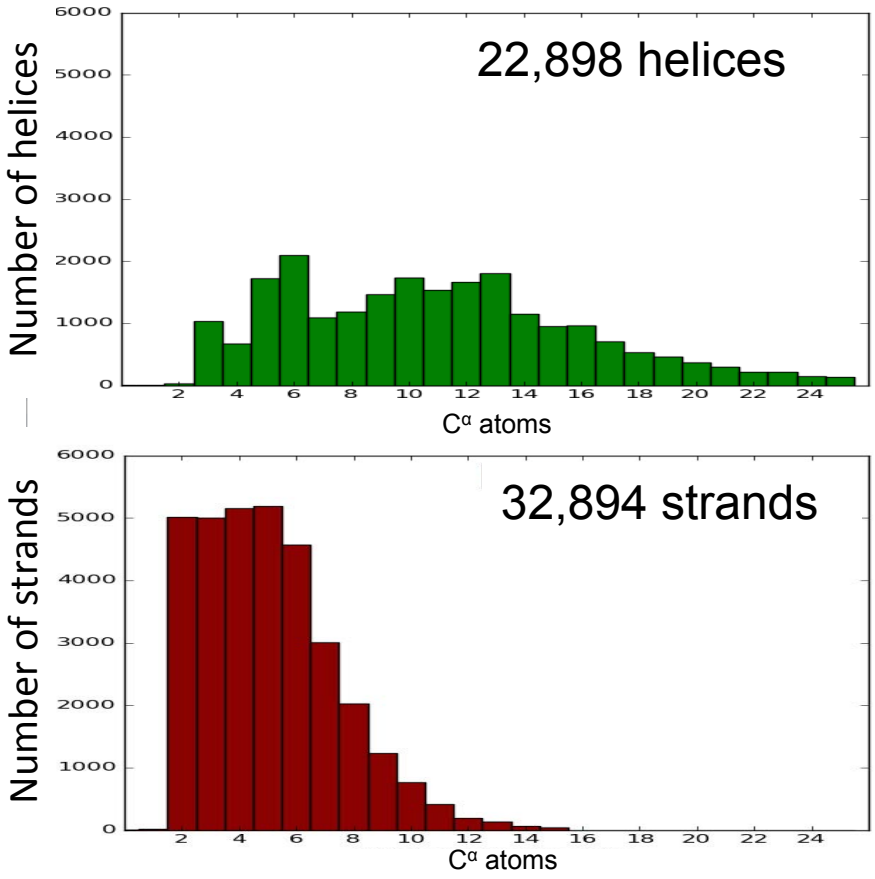


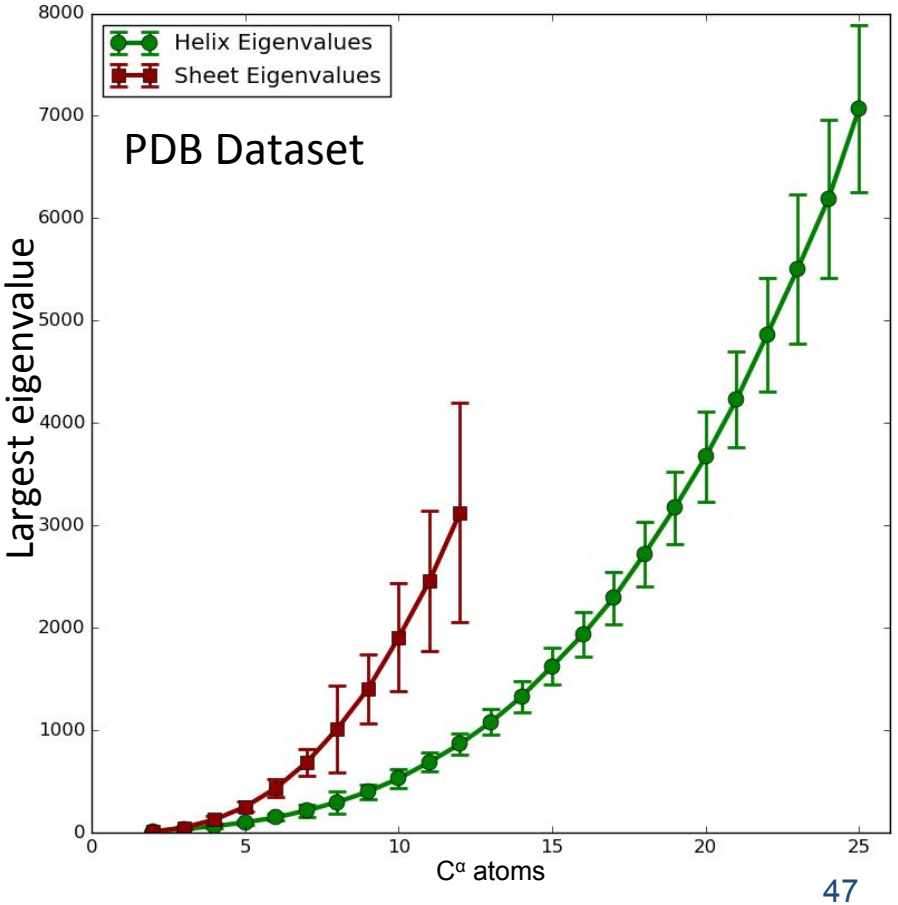
λ_{55} λ_{60} λ_{65} λ_{70} λ_{75} λ_{85}

Yes, the distance between two max eigenvalues serves as a proxy for distance between the two associated conformations!

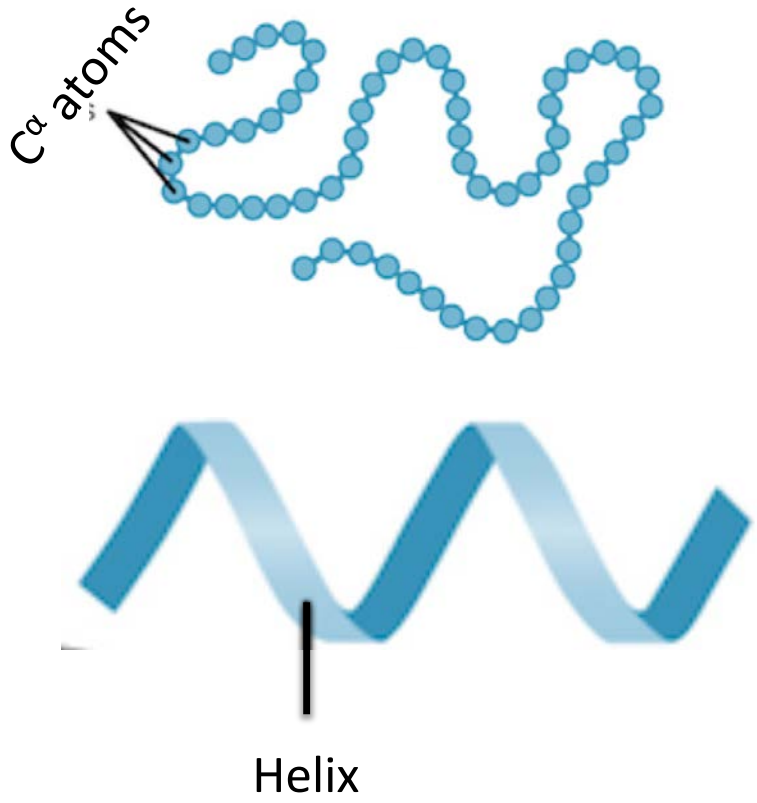
Mapping Largest Eigenvalues to Structures

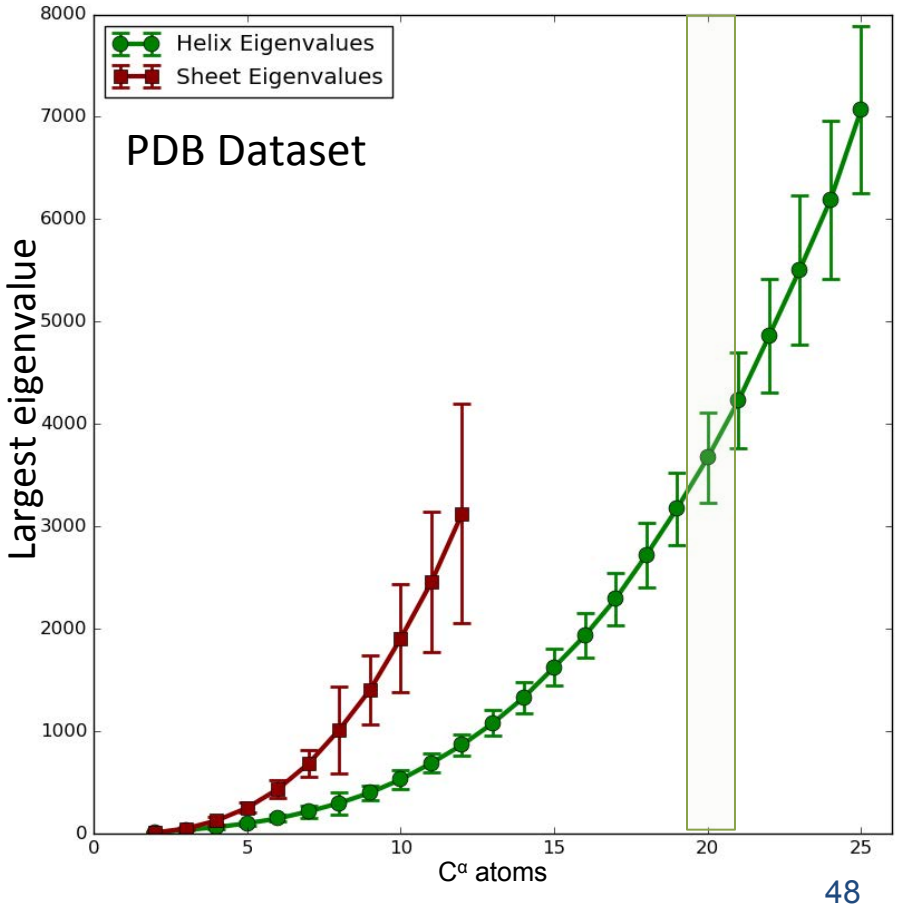
PDB dataset: 3,197 different proteins including 22,898 helices and 32,894 strands



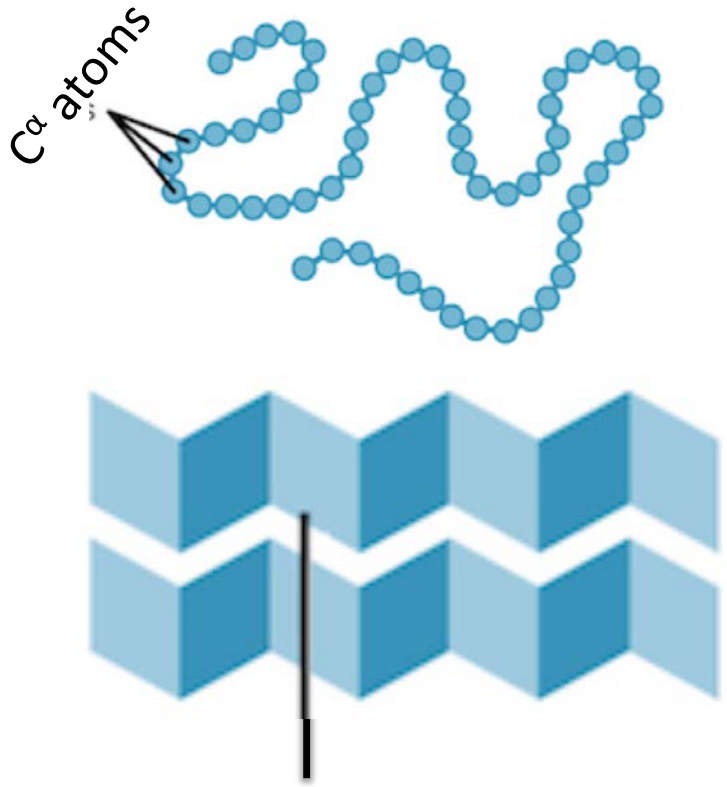


Mapping Largest Eigenvalues to Structures

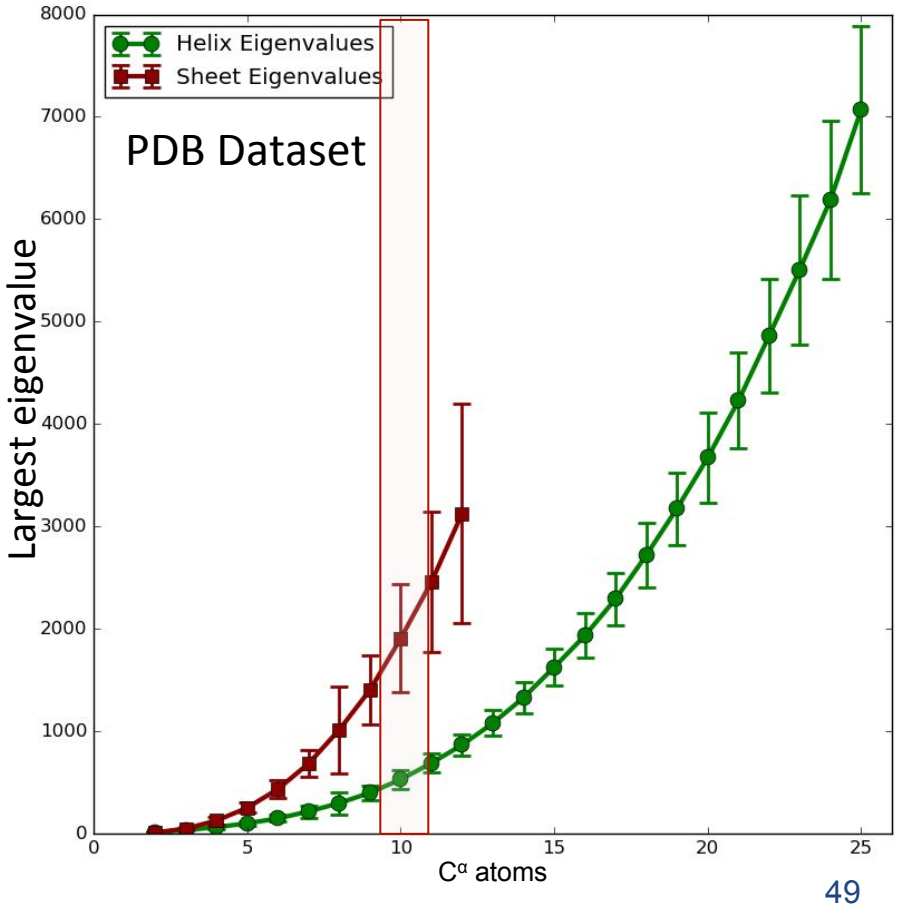




Mapping Largest Eigenvalues to Structures

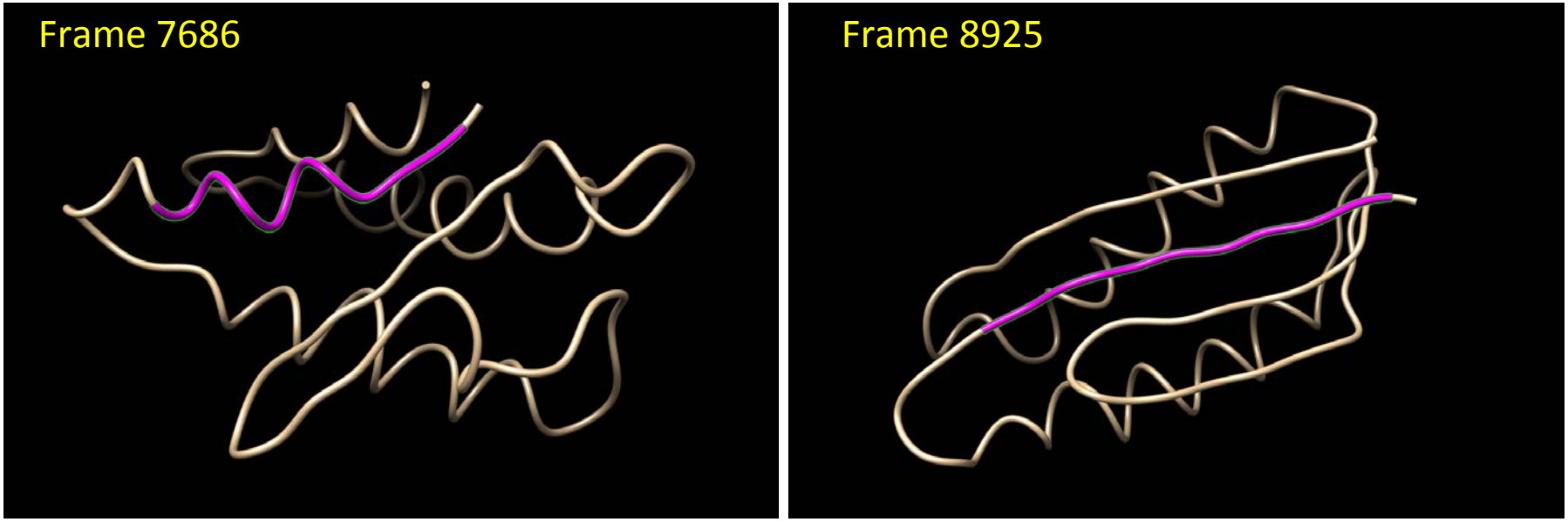


Sheets



Case Study I: 2MQ8 Protein

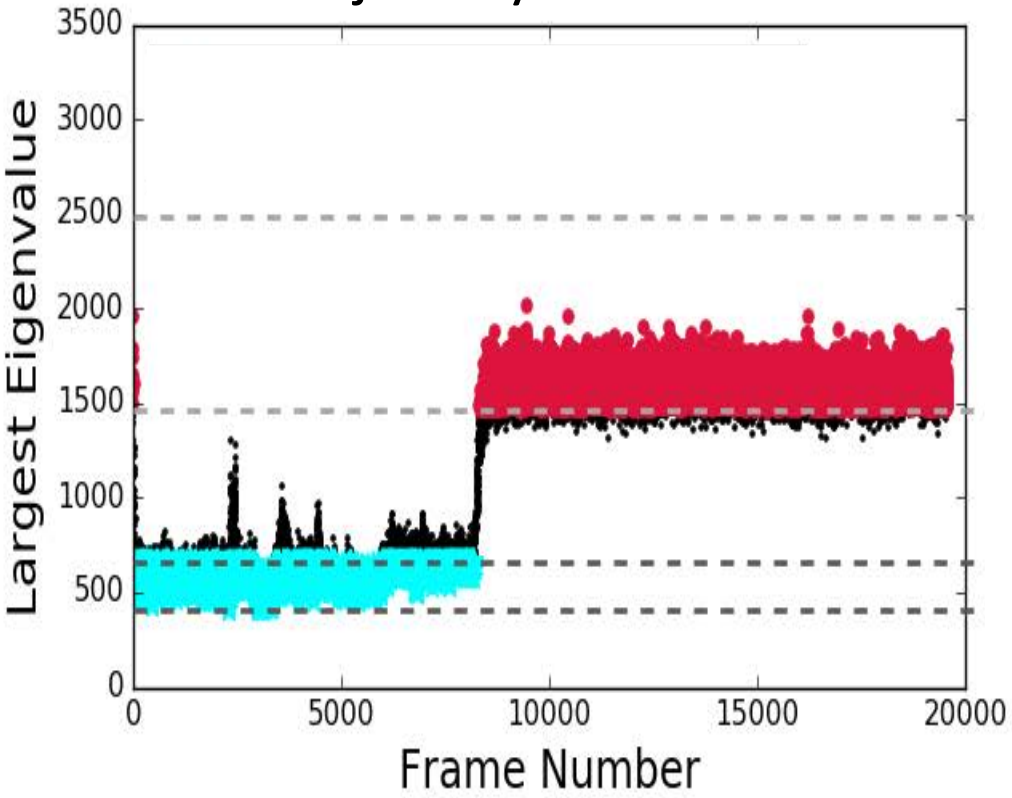
- Canonical simulation of 2MQ8 protein including both α helices and β strands
  - After ~9M steps α helices pack tighter and change into β strands



Can the eigenvalue analysis capture the conformational change?

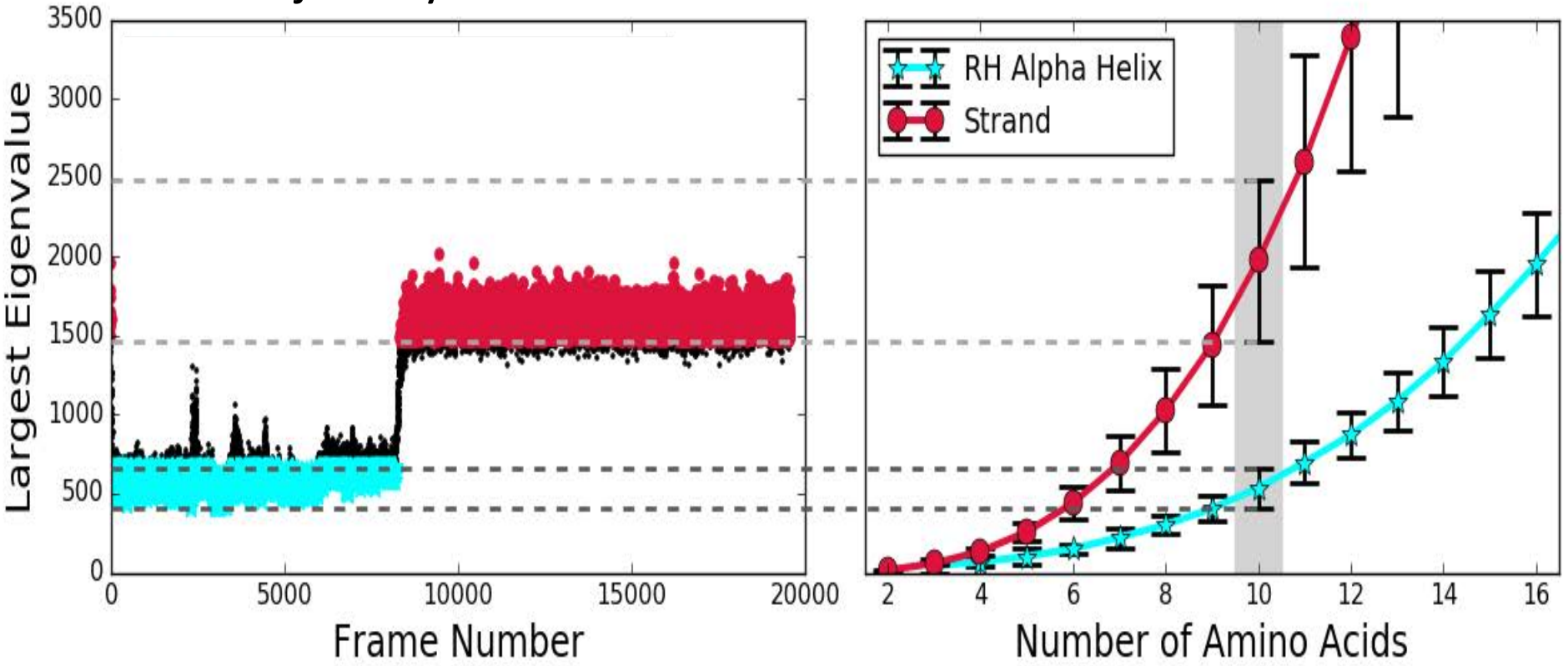
Case Study I: 2MQ8 Protein

Compute largest eigenvalue of 3rd strand (10 amino acids) for each trajectory frame



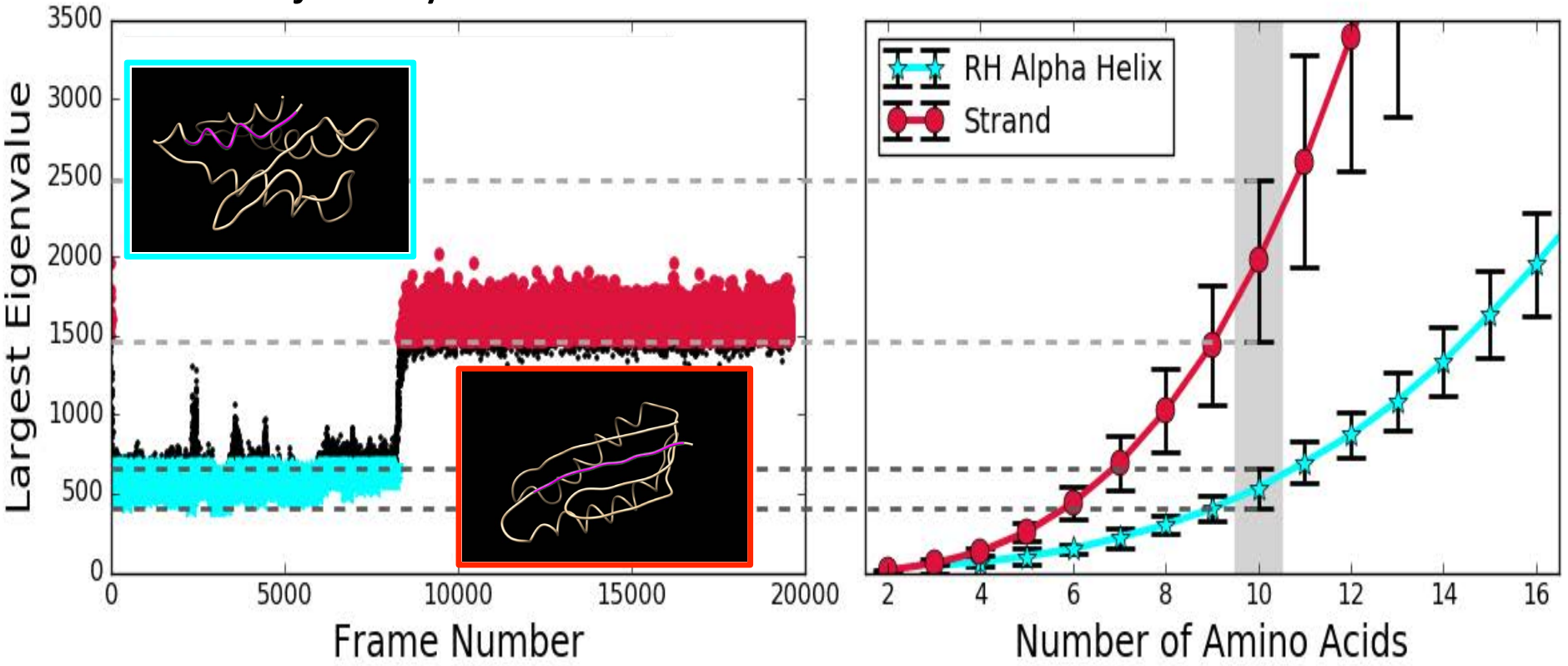
Case Study I: 2MQ8 Protein

Compute largest eigenvalue of 3rd strand (**10 amino acids**) for each trajectory frame



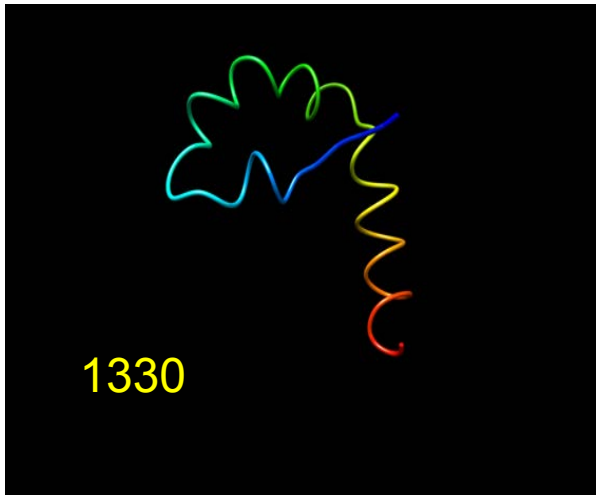
Case Study I: 2MQ8 Protein

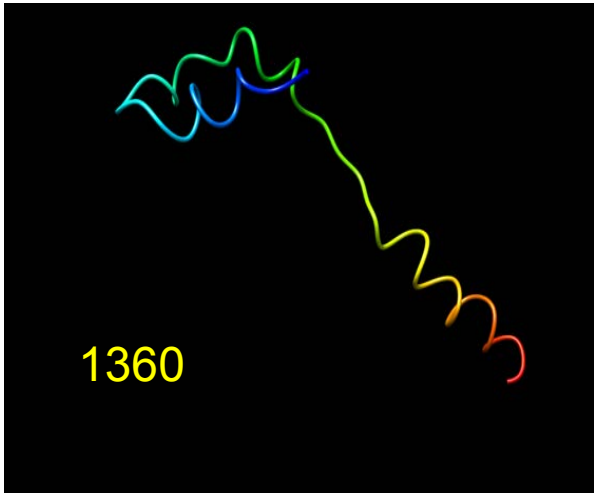
Compute largest eigenvalue of 3rd strand (10 amino acids) for each trajectory frame

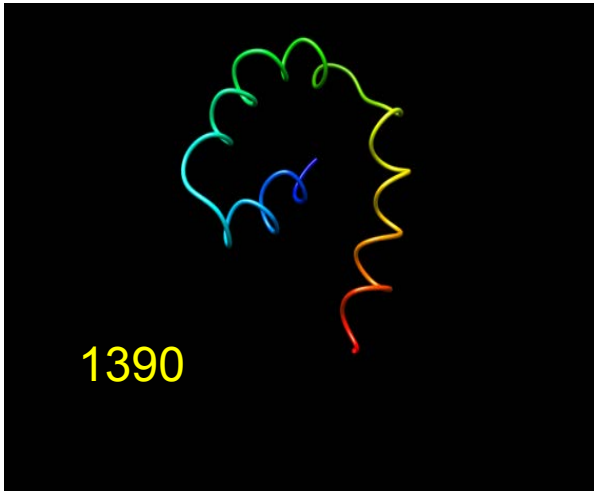


Case Study II: Capturing Movement of α-helices

Capture movement of structures with respect to each other

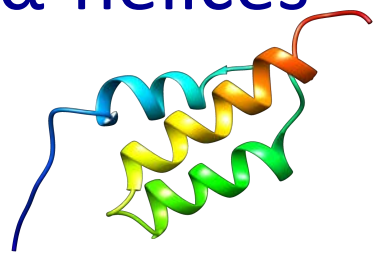




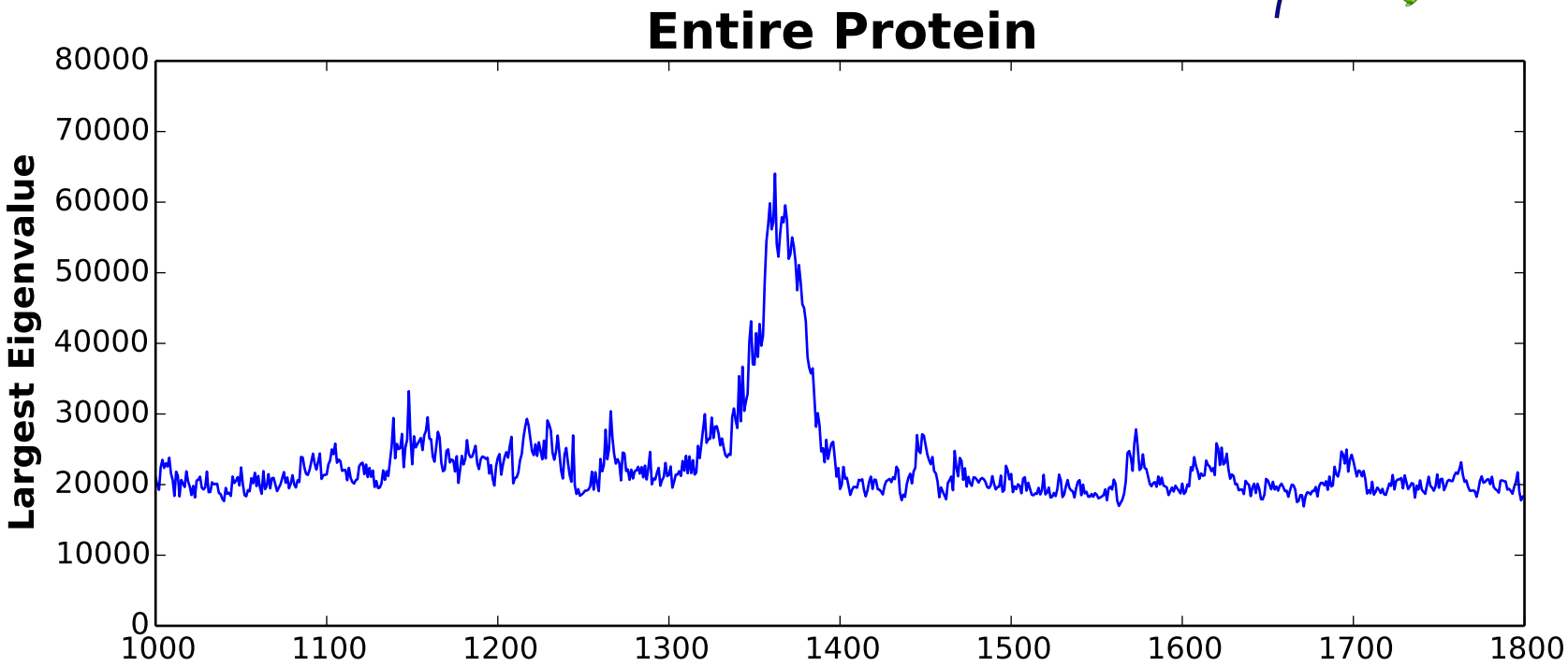


Can the eigenvalue analysis capture the movement of helices ?

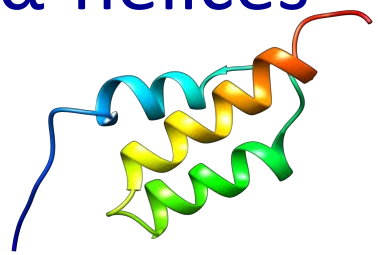
Case Study II: Capturing Movement of α-helices



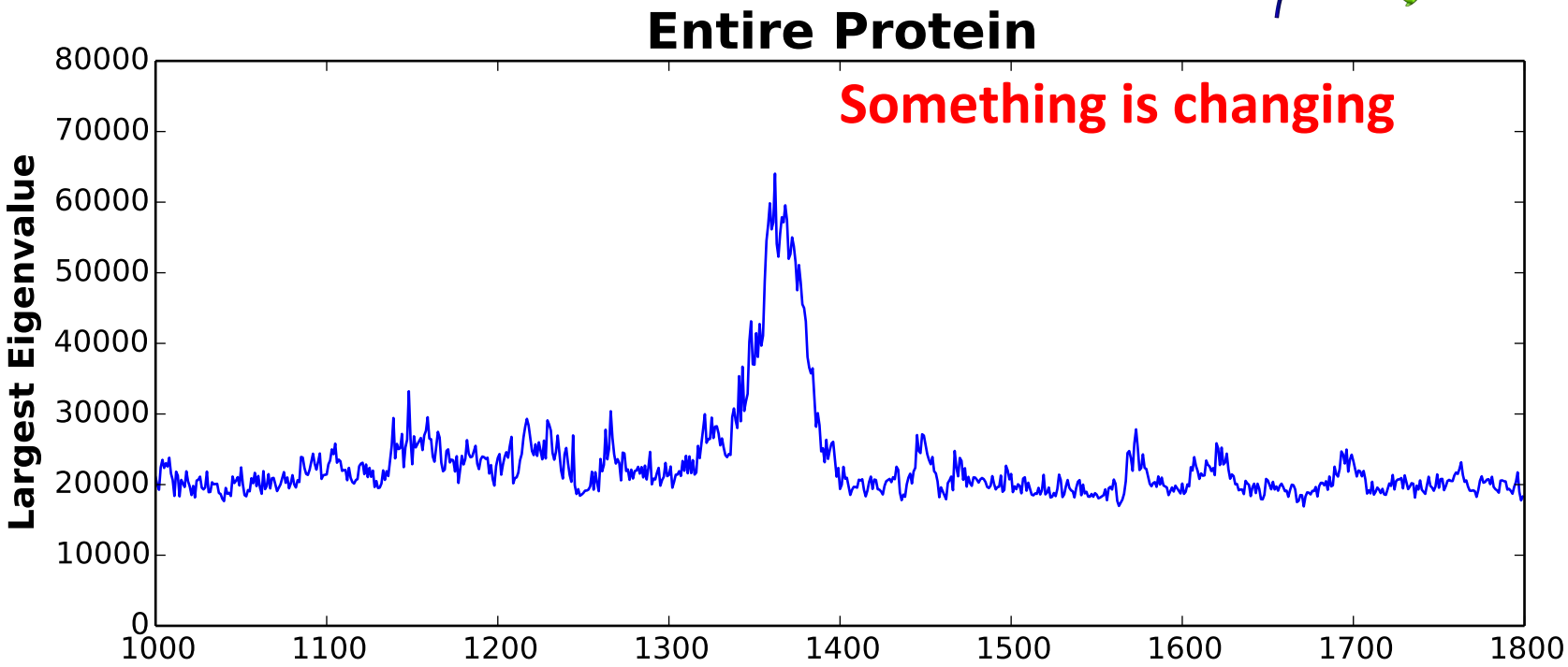
Monitor largest eigenvalue of entire protein



Case Study II: Capturing Movement of α-helices

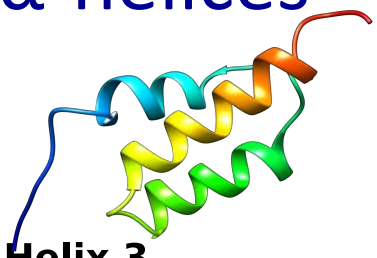


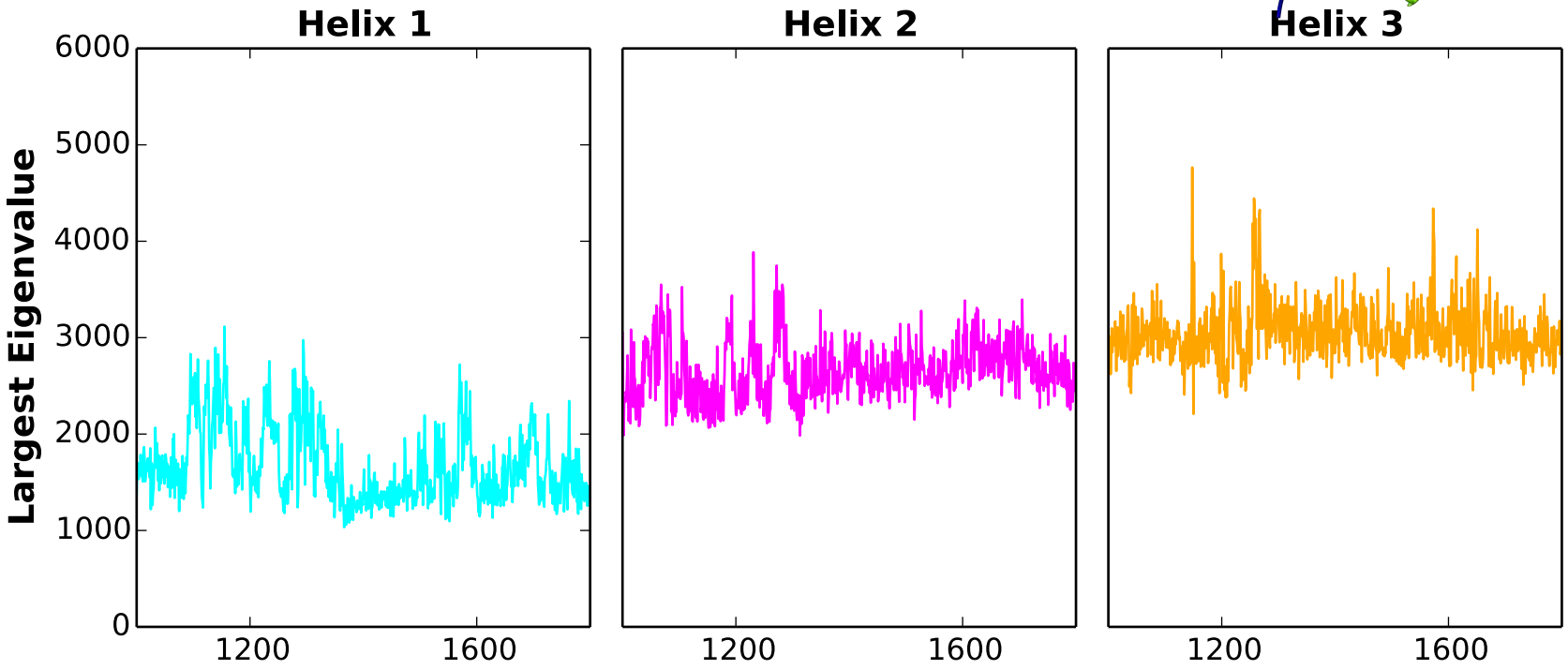
Monitor largest eigenvalue of entire protein



Case Study II: Capturing Movement of α-helices

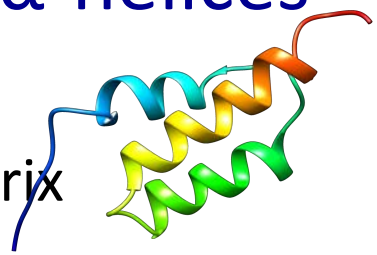
Monitor largest eigenvalue of single helices



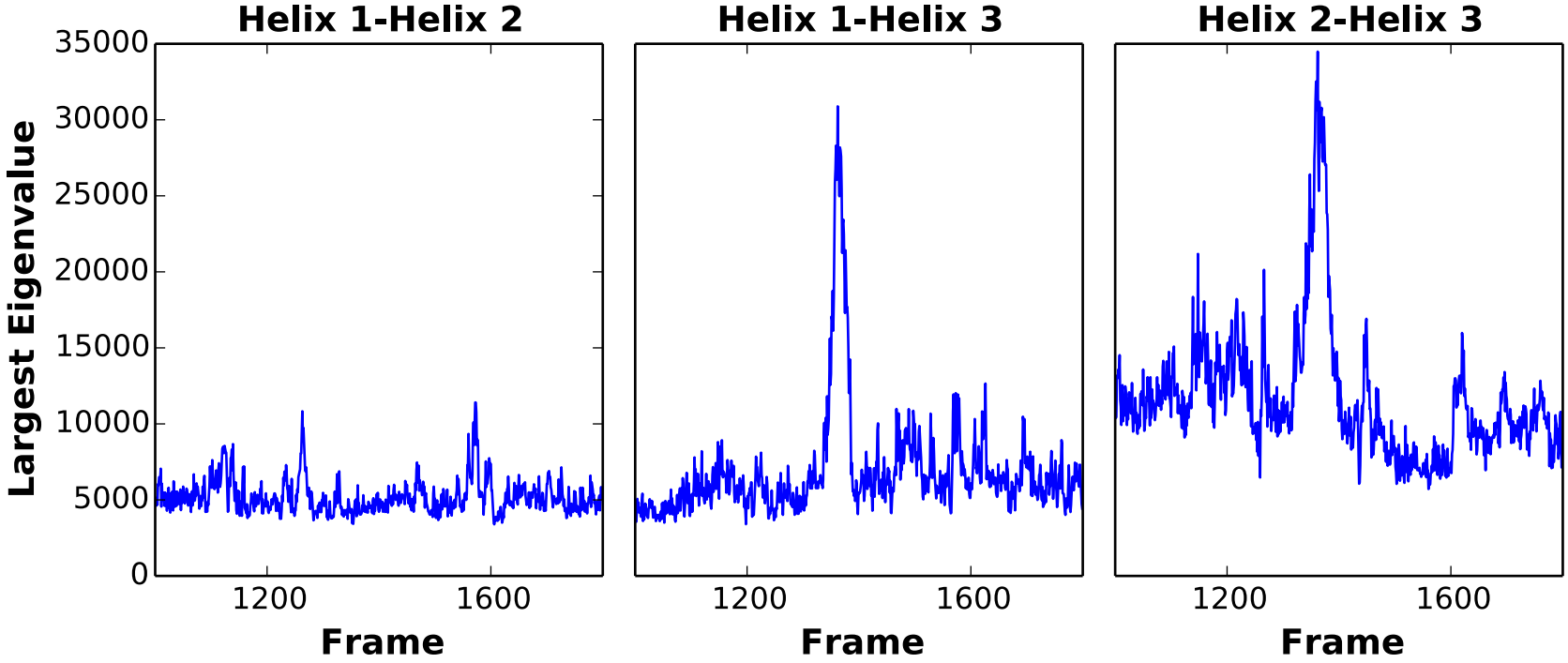


Individual α-helices (Helix 1, Helix 2, and Helix 3) appear stable

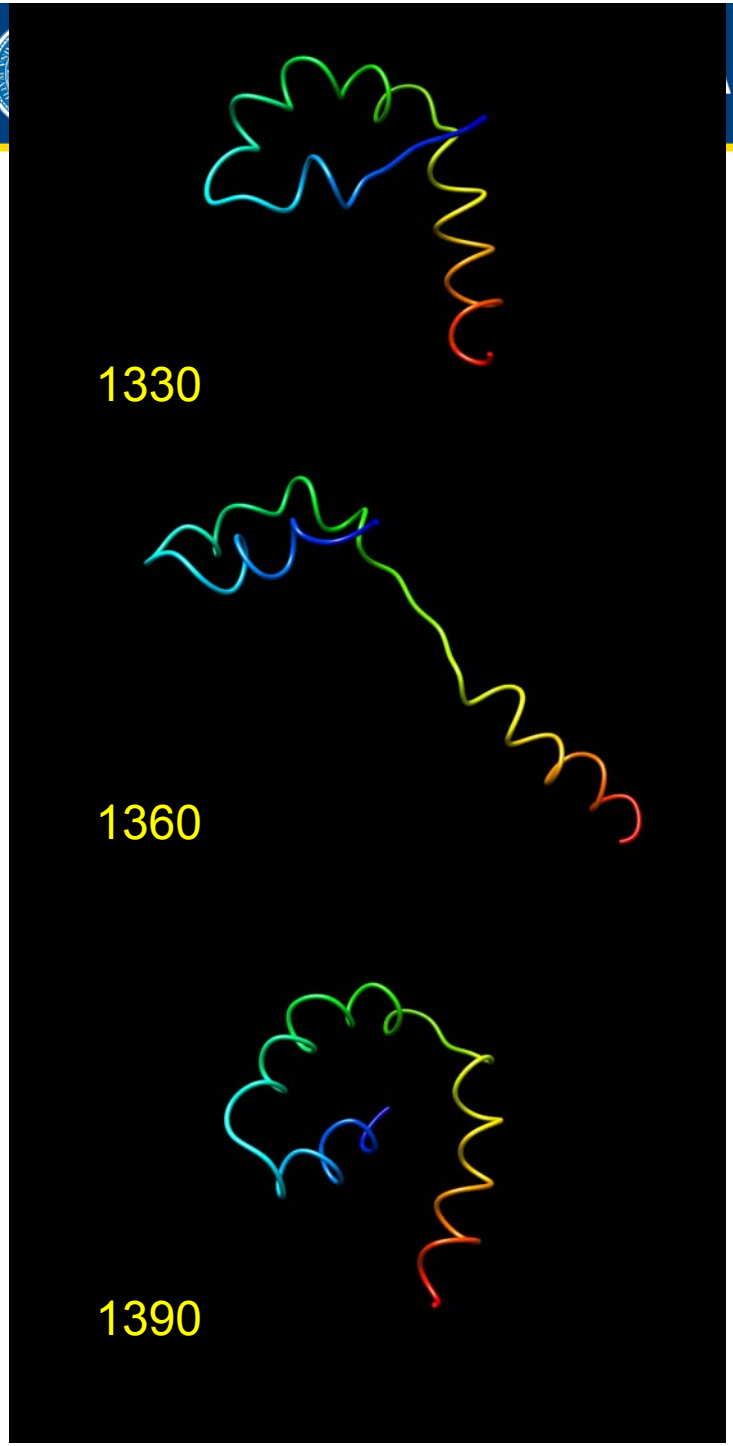
Case Study II: Capturing Movement of α-helices



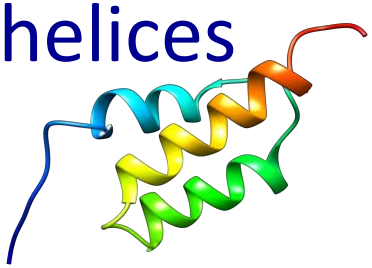
Monitor largest eigenvalue of bipartite distance matrix

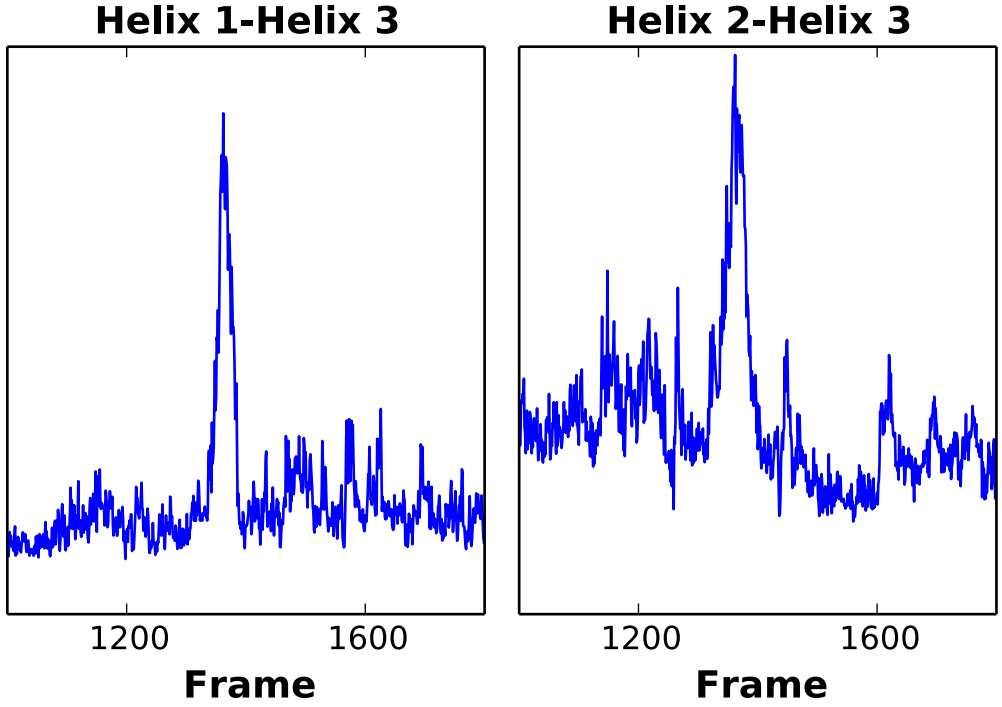


First and second α-helices appear stable; third helix moves



Case Study II: Capturing Movement of α-helices





Large relative change between two pairs of α-helices

"Storage technologies are advancing […] and it is really not clear at all [to me] that especially distributed storage platforms would not be able to handle […] petabyte data sets"

Anonymous Feedback

Yes, new technologies will be able to handle data at the extreme scale but *only* if we integrate new software paradigms.
I/O-aware schedulers are *a must*!
In-situ and in-transit analysis are *here to stay*!