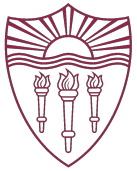


Enabling Efficient Execution of In Situ Workflows

Tu Mai Anh Do
Information Sciences Institute, University of Southern California

UTK Seminar, September 3rd, 2021

analytics for molecular dynamics

Acknowledgements


M. Taufer


E. Deelman


R. Ferreira da Silva


H. Weinstein


M. Cuendet


T. Estrada


S. Caino-Lores


L. Pottier


E. Kots


I. Lumsden

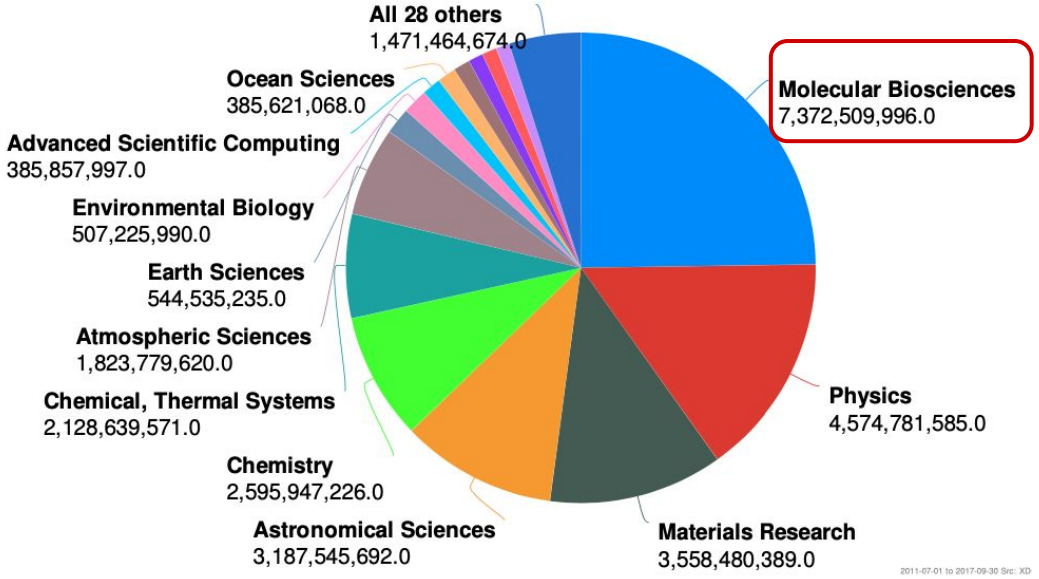
This work is funded by NSF contracts #1741040 and #1741057; and DOE contract #DE-SC0012636

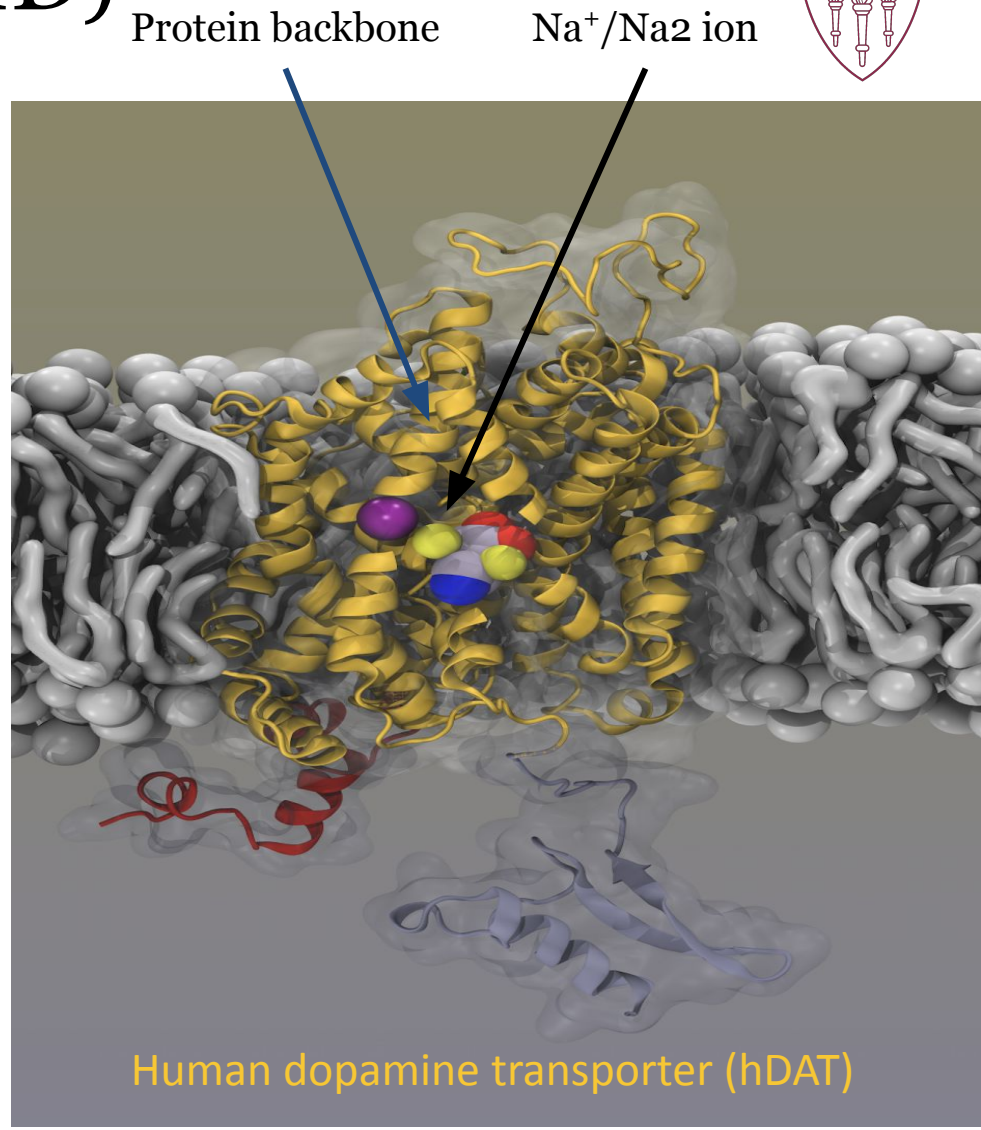
Outline

1. Motivations and Background
2. Contributions :
   a. Modeling framework for in situ workflows
   b. Performance indicators to determine performance of workflow ensembles
3. Conclusion

Molecular dynamics (MD)

Utilization breakdown on XSEDE machines for the period 2011-07 to 2017-09. (Simakov et al., 2018)



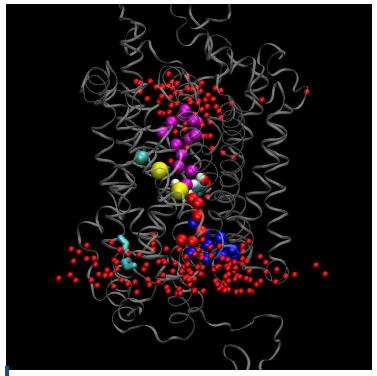


Razavi et al, 2017

- Molecular dynamics (MD) is a simulation model computing the atomic states of a molecular system evolving over time by observing interactions between atoms
- MD serves as a productive method to:
  - Control the configurations of the molecular systems
  - Observe important processes at atomic resolution

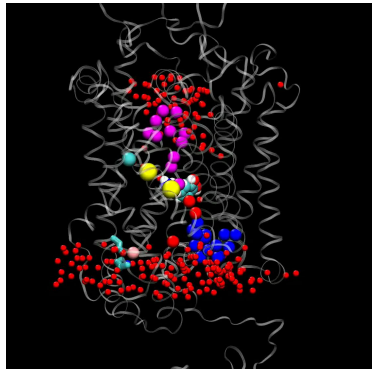
Traditional MD analysis

Initial Structure



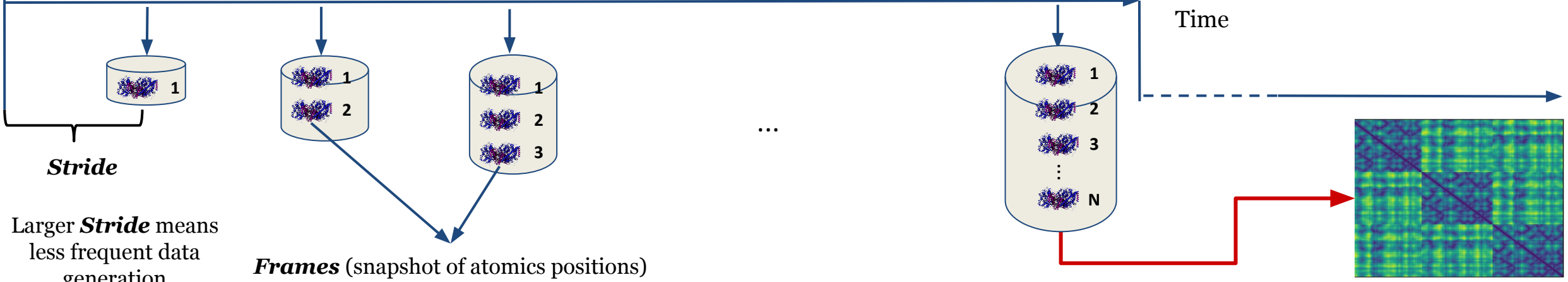
Changing Structure

Final Structure



need to follow dynamics by analyzing the trajectories

Simulated Time



Stride

Larger ***Stride*** means less frequent data generation

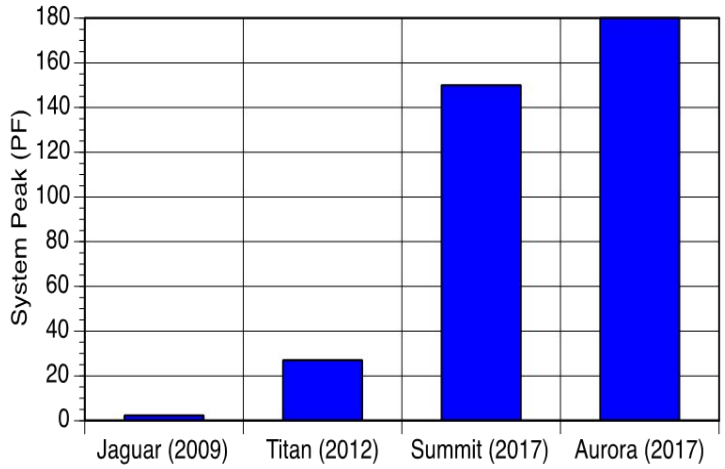
Frames (snapshot of atomics positions)

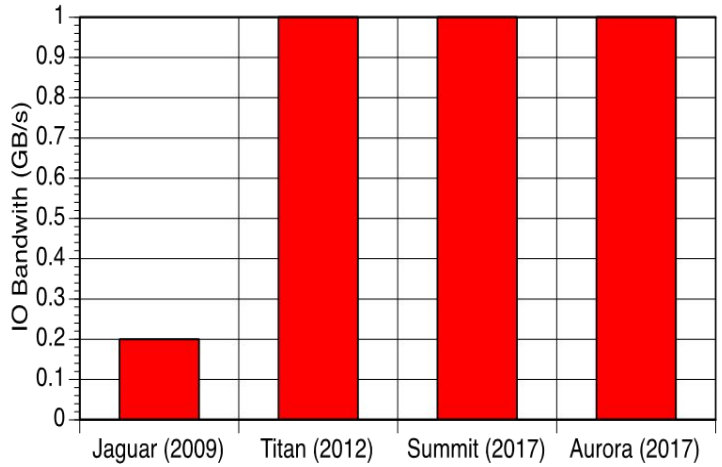
Post-hoc Analysis

I/O limitations

I/O stagnant on contemporary leadership computers. (Johnston et al., 2017)

Increasing - - →

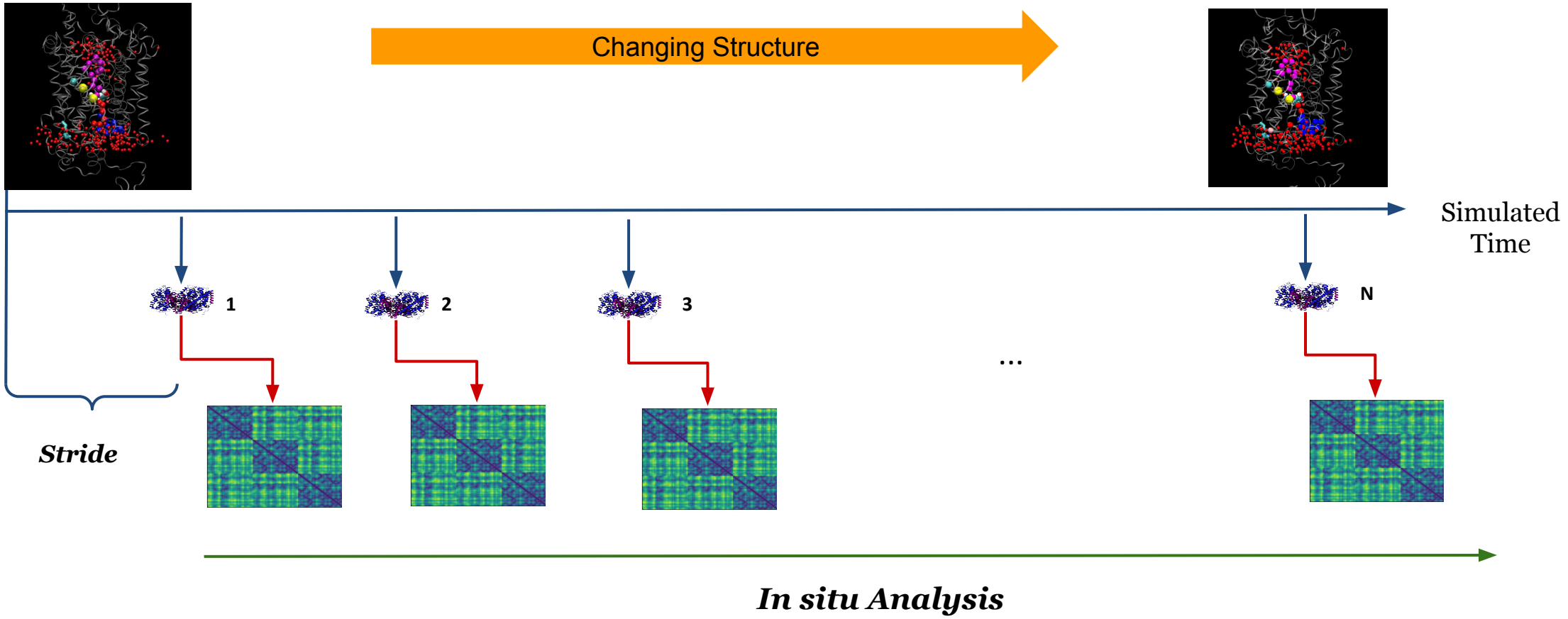




← - - - *Stagnant*

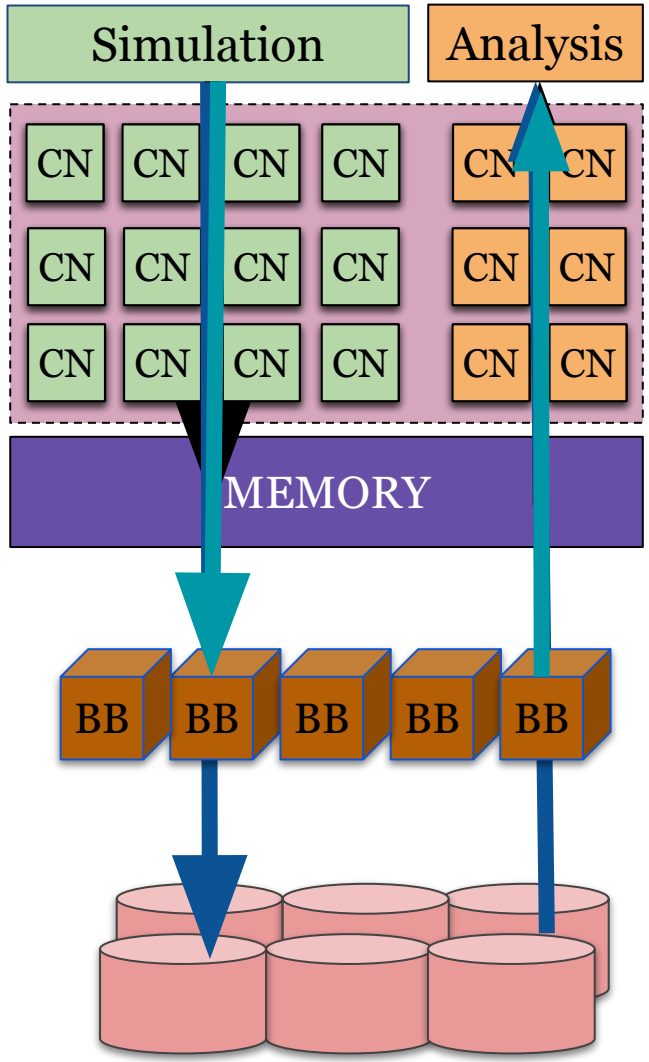
- The increase in computing capability helps the MD simulations generate more data that needs to be analyzed (150,000 atoms + 500,000 snapshots would generate ~ 1.8TB data)
- However, the I/O bandwidth does not grow at the same pace → I/O bottleneck

In situ analysis



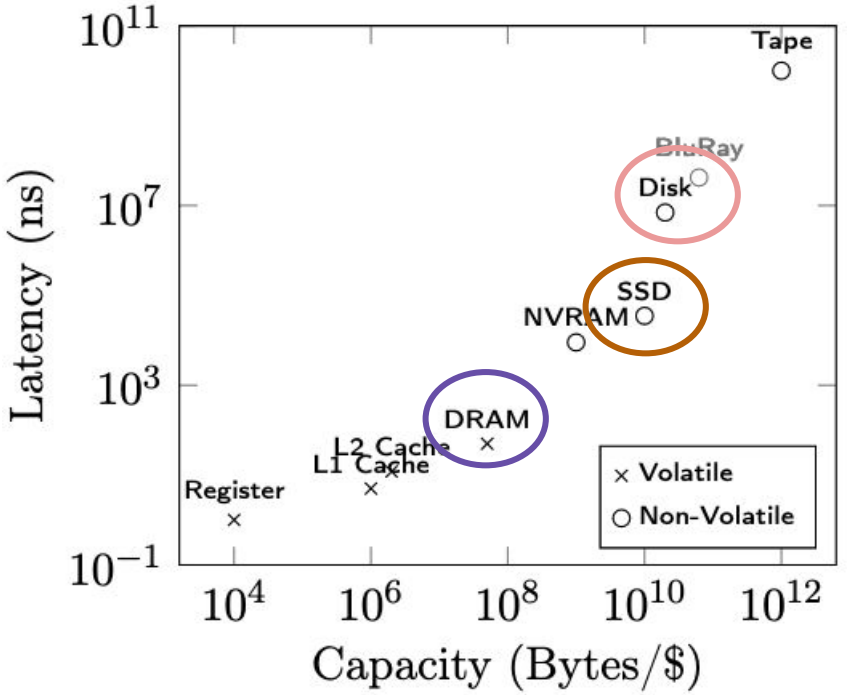
- Data is analyzed as soon as generated
- Decoupling analysis from the simulation to interleave their executions → Reduce time-to-solution
- Study insights into phenomena of the molecular system in a timely fashion

Data staging



Burst buffers

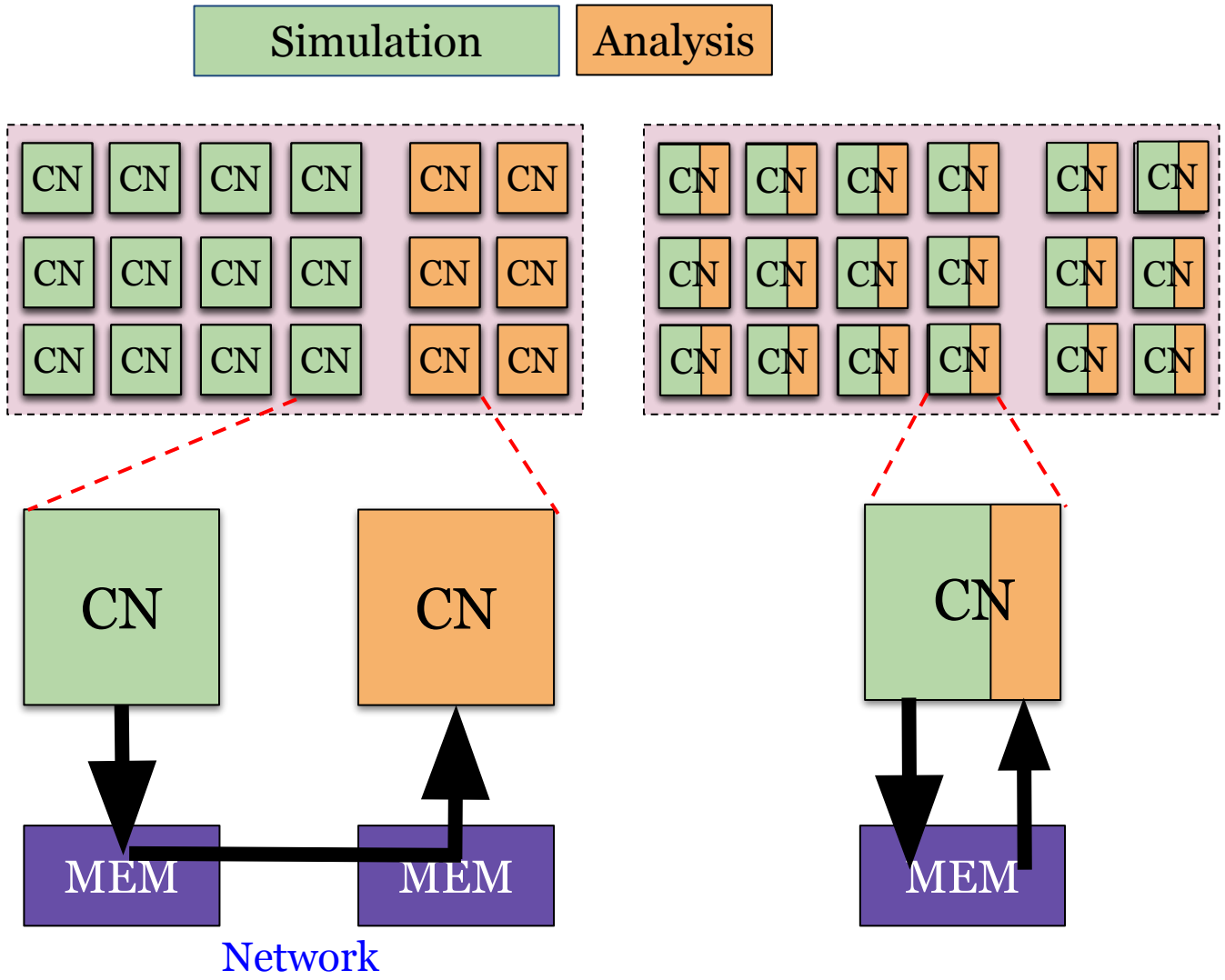
Parallel File System



Comparison of capacity, latency for most important technologies of the memory and storage hierarchies. (Lüttgau et al., 2018)

- We focus on memory-to-memory for the lowest latency

Placement variants



In transit
Dedicated resources
Increase data movement cost

Co-location (Helper-core)
Resource contention
Increases data locality

In situ framework design

Problem: How to enable in situ execution of simulations and analyses?

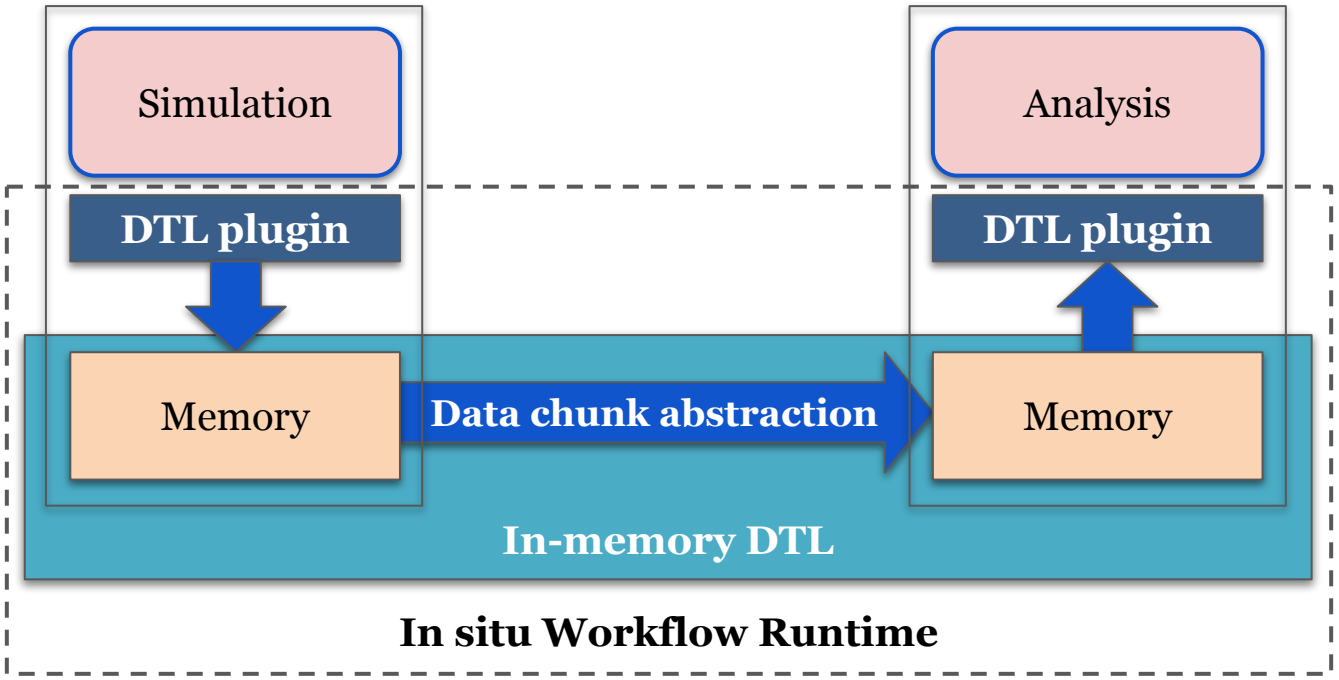
Challenges/complexities:

- Decoupling analyses from the simulation to perform in situ requires to manage data staging to additional concurrent components
- Orchestrating data coupling over iterations
- Data incompatibility between decoupled components (simulations, analyses)

We model/design a framework that allows to decouple in situ analyses from the simulation to address these complexities:

- Data staging, coupling → Data transport layer
- Data compatibility → Data abstraction

In situ framework implementation



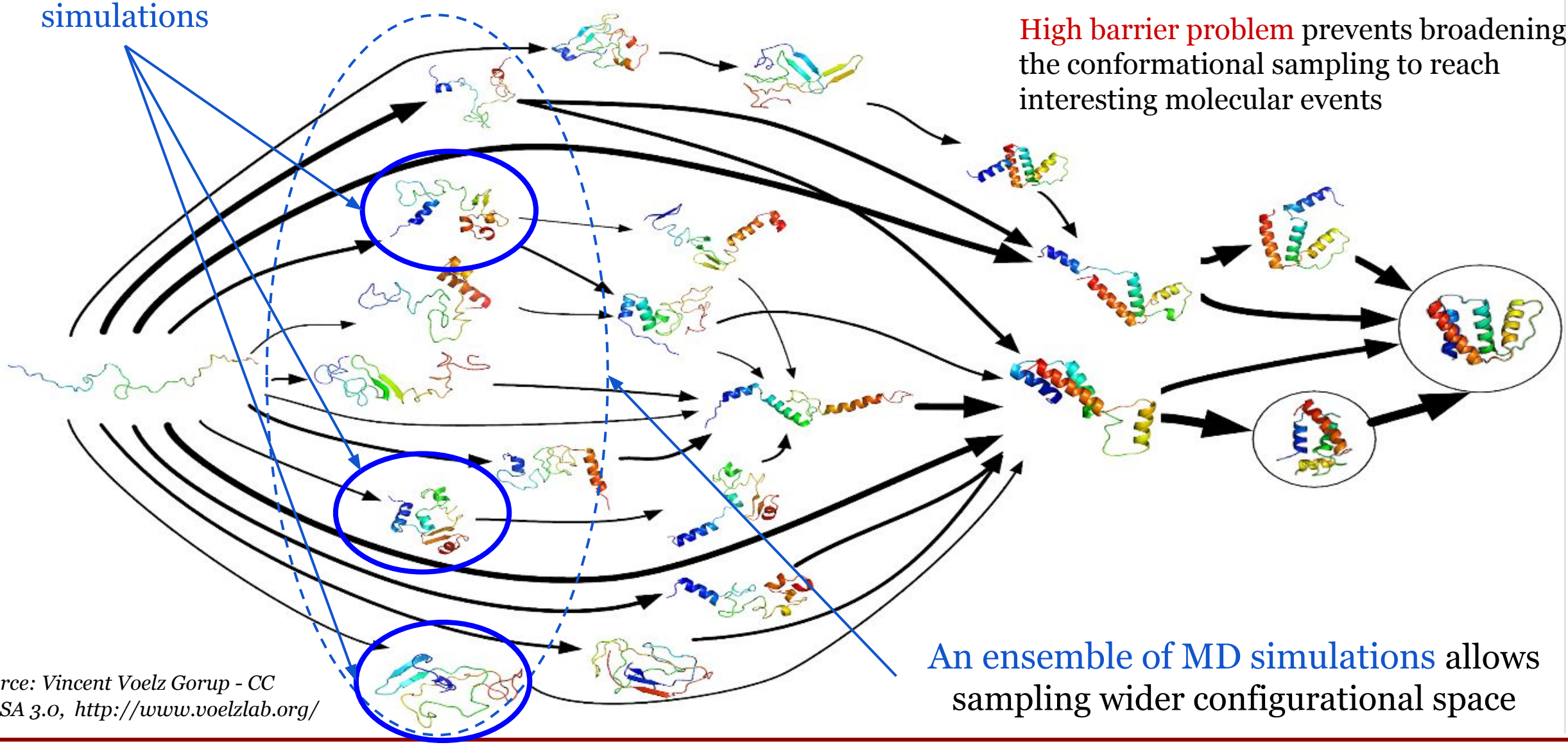
- Two main components:
  - Data transport layer (DTL) provides interfaces to support different storage tiers
  - DTL plugins interact with the DTL via data chunk abstraction for compatibility

- In this work
  - The DTL is implemented with the help of DIMES (Zhang et al., 2017) to enable in-memory data staging
  - The DTL plugin is integrated with Plumed (Tribello et al., 2014) to offer non-intrusive approach

MD simulation ensemble

MD short simulations

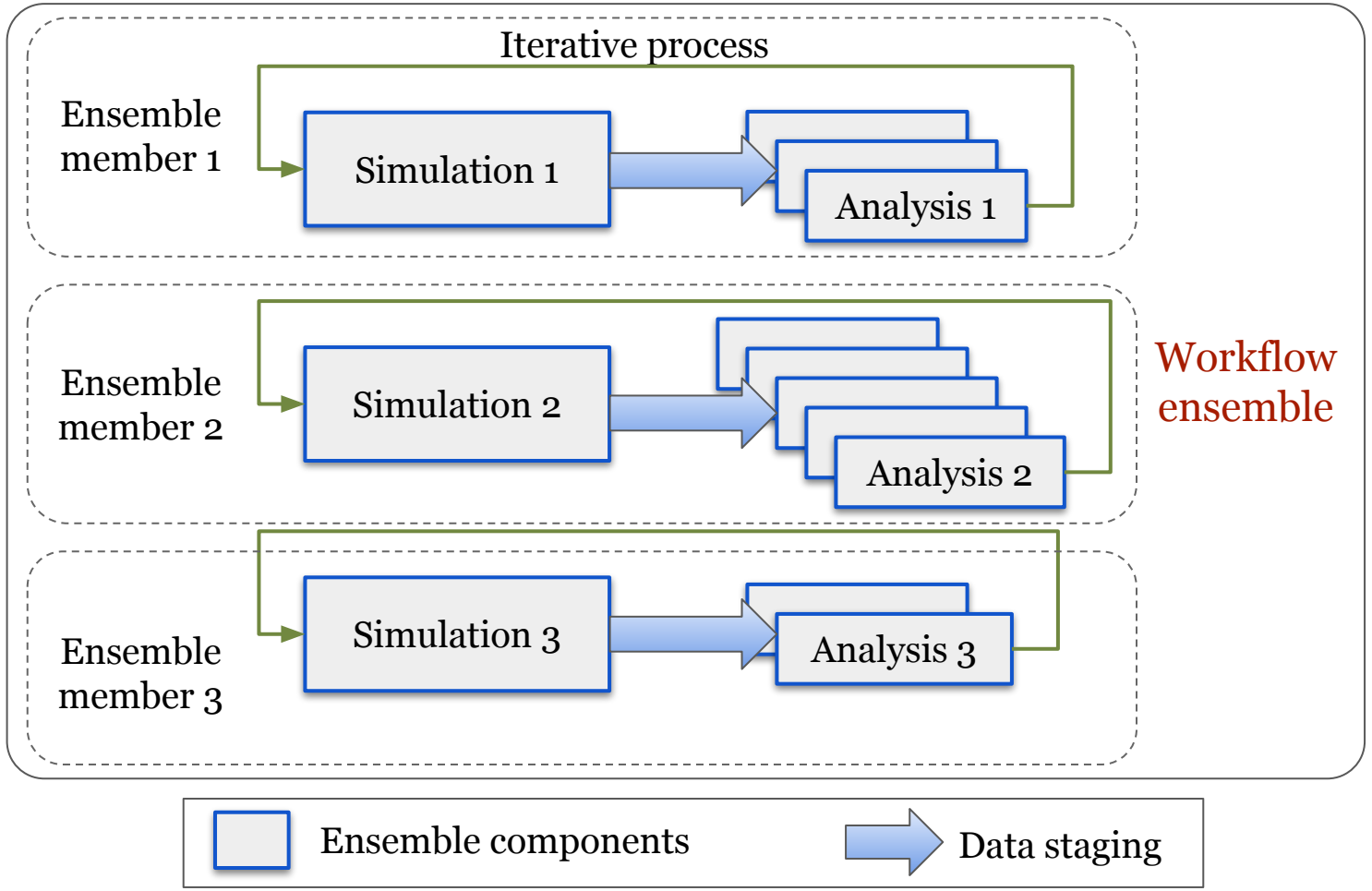
High barrier problem prevents broadening the conformational sampling to reach interesting molecular events



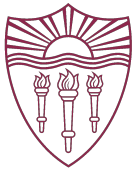
Source: Vincent Voelz Gorup - CC BY-SA 3.0, http://www.voelzlab.org/

An ensemble of MD simulations allows sampling wider configurational space

In situ Workflow Ensembles



Performance indicators for in situ workflow ensembles



Problem: How to evaluate performance of an in situ workflow ensembles?

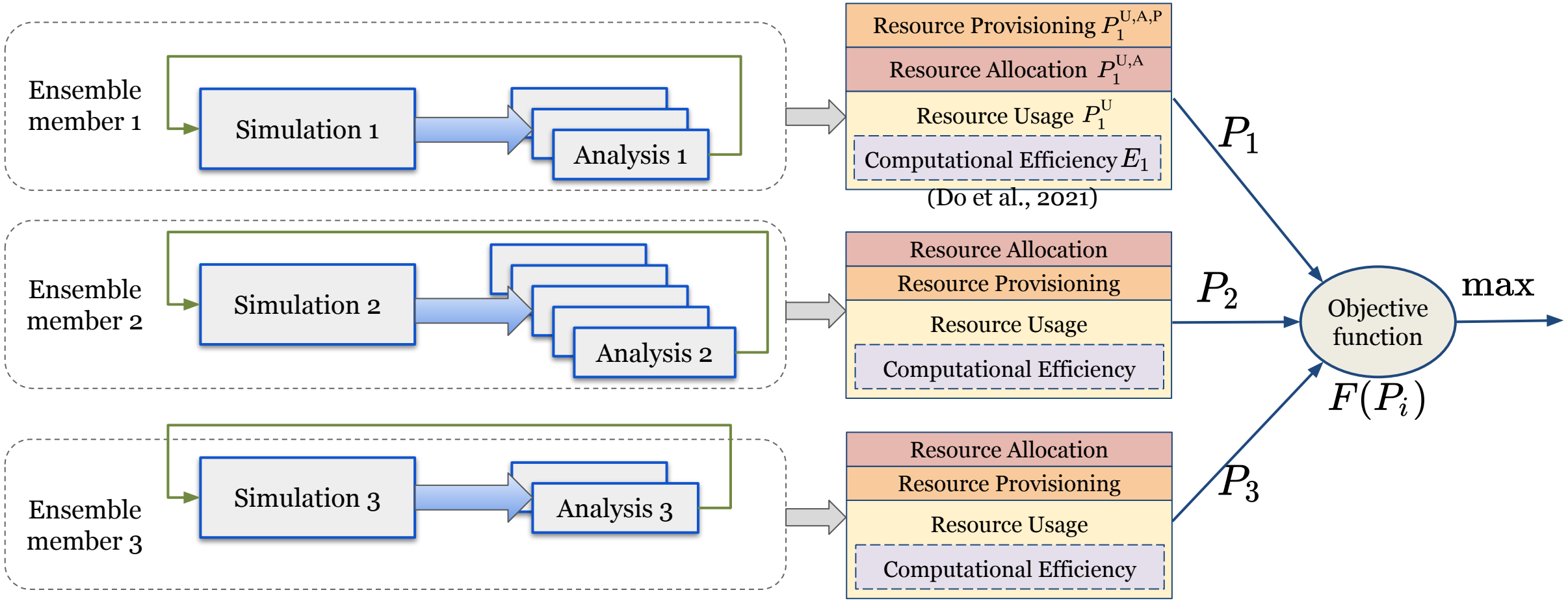
Challenges:

- Using traditional metrics and capturing them separately are not straightforward to synthesize performance of entire workflow ensemble
- Need to take into account resources to characterize performance under resource cost

We introduce multi-stage performance indicators that capture performance of the entire in situ workflow ensembles in terms of multiple resource perspectives

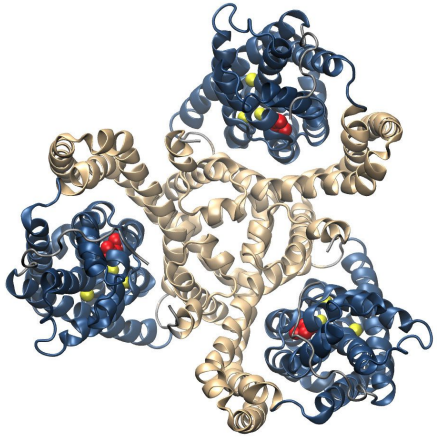
- Resource usage: How efficiently the resource is utilized?
- Resource allocation: How efficiently the simulations, analyses are placed on allocated resources?
- Resource provisioning: How many resources are provisioned to execute efficiently?

Multi-stage performance indicators



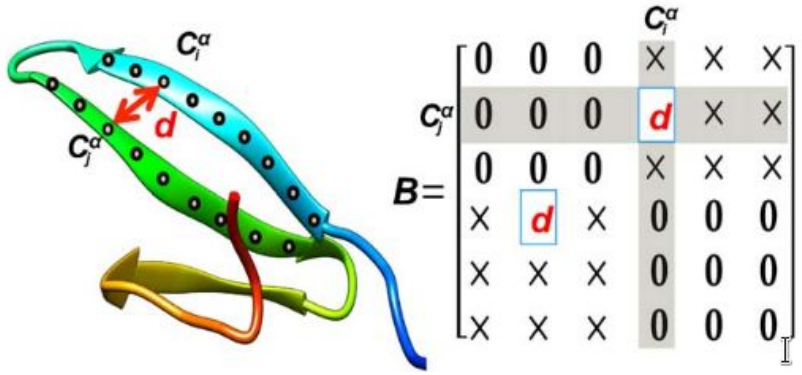
Experiment setup

Simulations



Medium-scale all-atom system containing the GltPh transporter protein (Akyuz 2015) implemented in GROMACS (P Bjelkmar et al., 2010)

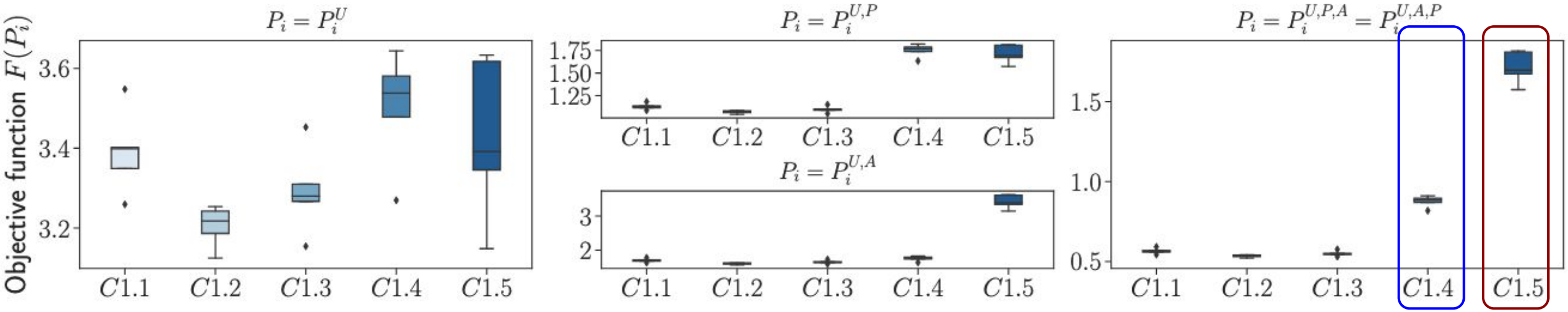
Analyses



Collective variable (largest eigenvalue of bipartite distance matrices between two substructures) (Barducci 2011, Johnston 2017)

- Execution platform: Cori, a Cray XC40 supercomputer at NERSC. Each compute node is equipped with
  - 2 Intel Xeon E5-2698 v3 (16 cores each)
  - 128 GB of DRAM
- TAU (Shende et al., 2006) is leveraged to collect performance information

One analysis per simulation



➔ $C1.5$ outperforms other configurations, which validates the benefit of co-locating coupled components

Config-uration	Number of computing nodes	Number of ensemble members	Node indexes			
			Ensemble member 1		Ensemble member 2	
			Simulation 1	Analysis 1	Simulation 2	Analysis 2
C_f	2	1	n_0	n_1	-	-
C_c	1	1	n_0	n_0	-	-
C1.1	3	2	n_0	n_2	n_1	n_2
C1.2	3	2	n_0	n_1	n_0	n_2
C1.3	3	2	n_0	n_0	n_1	n_2
⋆ C1.4	2	2	n_0	n_1	n_0	n_1
⋆⋆ C1.5	2	2	n_0	n_0	n_1	n_1

Takeaways

- The proposed indicators can be leveraged for evaluating scheduling decision of in situ ensemble under resource constraints
- Provide hints to improve effectiveness of resource usage → optimizing simulation exploration by running many MD simulations at a time
- Future work will consider leveraging the proposed indicators for scheduling in situ components of a workflow ensemble to enable high-throughput ensemble of simulations


Thank You!
Q&A

References

- Simakov et al., 2018. A Workload Analysis of NSF's Innovative HPC Resources Using XDMoD. arXiv preprint arXiv:1801.04306 (2018)
- Lüttgau et al., 2018. Survey of Storage Systems for High-Performance Computing. Supercomputing Frontiers And Innovations, 5(1), 31-58.
- Chelli et al., 2012. Serial Generalized Ensemble Simulations of Biomolecules with Self-Consistent Determination of Weights. Journal of Chemical Theory and Computation 8, 3.
- Sanchez-Martinez et al., 2015. Enzymatic minimum free energy path calculations using swarms of trajectories. J Phys Chem B. 2015;119(3):1103–13
- Ballard et al., 2009. Replica exchange with nonequilibrium switches. Proc Natl Acad Sci. 2009;106(30):12224–9
- Comer et al., 2014. Multiple-Replica Strategies for Free-Energy Calculations in NAMD: Multiple-Walker Adaptive Biasing Force and Walker Selection Rules. Journal of Chemical Theory and Computation 10, 12 (2014).
- Bowman et al., 2010. Enhanced Modeling via Network Theory: Adaptive Sampling of Markov State Models. Journal of Chemical Theory and Computation 2010 6 (3), 787-794
- P Bjelkmar et al., 2010. Implementation of the CHARMM Force Field in GROMACS:Analysis of Protein Stability Effects from Correction Maps, Virtual Interaction Sites, and Water Models. J. Chem. Theory Comput.6, 2 (2010).
- Tribello et al., 2014. Plumed 2: New feathers for an old bird. Comput. Phys. Commun., 185(2):604–613.

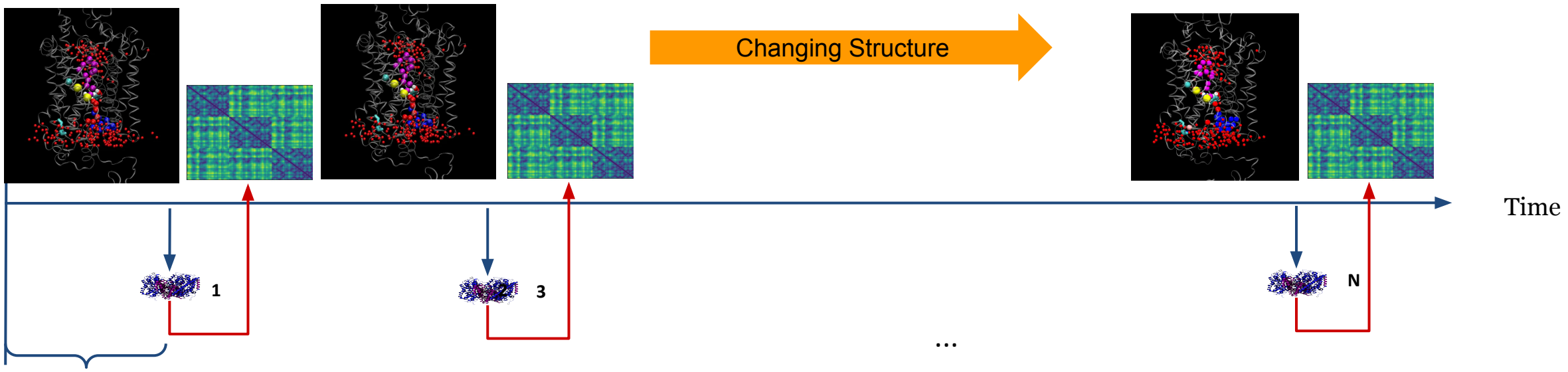
References

- Razavi et al, 2017. A Markov State-based Quantitative Kinetic Model of Sodium Release from the Dopamine Transporter. Sci Rep 7, 40076
- Pronk et al., 2011. Copernicus: a new paradigm for parallel adaptive molecular dynamics. In Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC '11). Association for Computing Machinery, New York, NY, USA, Article 60, 1–10.
- Hruska et al., 2020. Extensible and Scalable Adaptive Sampling on Supercomputers. Journal of Chemical Theory and Computation 2020 16 (12), 7915-7925
- Johnston et al. “In situ data analytics and indexing of protein trajectories”, Journal of Computational Chemistry, 38 (16), 1419-1430, (2017)
- Akyuz et al. 2015. Transport domain unlocking sets the uptake rate of an aspartate transporter. Nature 518, 7537(2015)
- Barducci et al. 2011. Meta-dynamics.WIREs Computational Molecular Science 1, 5 (2011)
- Johnston et al. In situ data analytics and indexing of protein trajectories. Journal of Computational Chemistry, 38 (16), 1419-1430, (2017)
- Fan Zhang et al., 2017. In-memory staging and data-centric task placement for coupled scientific simulation workflows.Concurrency and Computation: Practice and Experience 29, 12 (2017)
- Agelastos et al., 2014. The Lightweight Distributed Metric Service: A Scalable Infrastructure for Continuous Monitoring of Large Scale Computing Systems and Applications. International Conference for High Performance Computing, Networking, Storage and Analysis (SC ’14). 154–165.

References

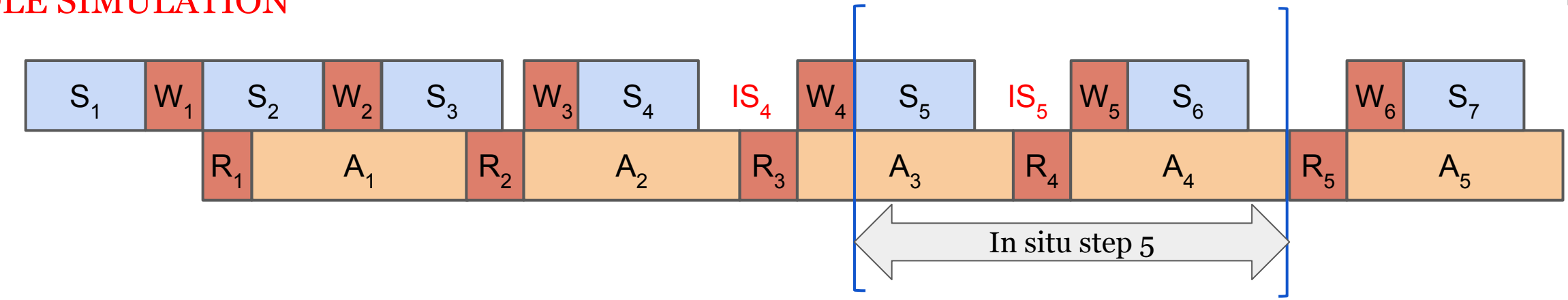
- Chad Wood et al., 2016. A Scalable Observation System for Introspection and in Situ Analytics. The 5th Workshop on Extreme-Scale Programming Tools (ESPT '16). IEEE Press, Salt Lake City, Utah, 42–49.
- Taufer et al., 2019.Characterizing In Situ and In Transit Analytics of Molecular Dynamics Simulations for Next-Generation Supercomputers. 15th International Conference on eScience (eScience)
- Zou et al., 2014. FlexAnalytics: A flexible data analytics framework for big data applications with I/O performance improvement. Big Data Research 1, 4 – 13.
- Zhang et al., 2016. WOWMON: A Machine Learning-based Profiler for Self-adaptive Instrumentation of Scientific Workflows. Procedia Computer Science 80 (2016), 1507–1518. International Conference on Computational Science 2016, ICCS.
- Malakar et al., 2015. Optimal scheduling of in-situ analysis for large-scale scientific simulations. International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-11
- Shende et al., 2006. The Tau parallel performance system. The International Journal of High Performance Computing Applications 20, 287–311.
- Lofstead et al., 2008. Flexible io and integration for scientific codes through the adaptable io system (adios),6th international workshop on Challenges of large applications in distributed environments
- Docan et al., 2012.Dataspaces: an interaction and coordination framework for coupled simulation workflows. Cluster Computing .
- Dreher, M., et al., 2017. Decaf: Decoupled dataflows for in situ high-performance workflows

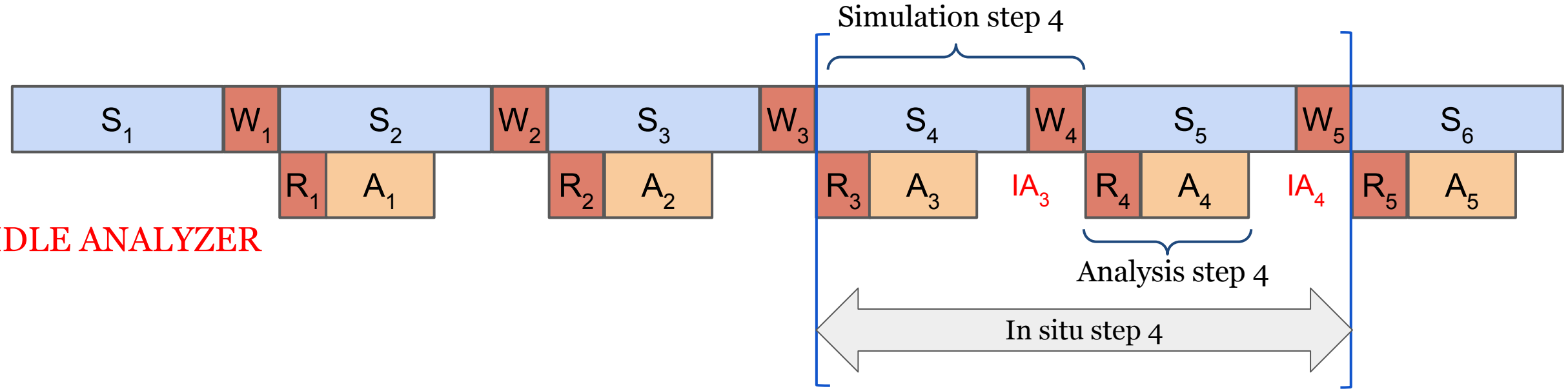
In-line in situ analysis



- Data is analyzed as soon as generated
- The simulation and analysis interchangeably execute
- Analysis needs to be embedded in simulation code
- Less robust

IDLE SIMULATION





IDLE ANALYZER

•

Efficiency model

- Computing in situ step is lightweight and can be performed online
- *In situ step* is leveraged to estimate:
  1. Makespan: duration of an in situ step x number of in situ steps - overlapped part between in situ steps
  2. Useful computation: measured by time for computation except idle time during an in situ step
- Evaluate computational efficiency:

Maximize $$E = \frac{\text{Estimation of useful computation}}{\text{Estimation of makespan}}$$

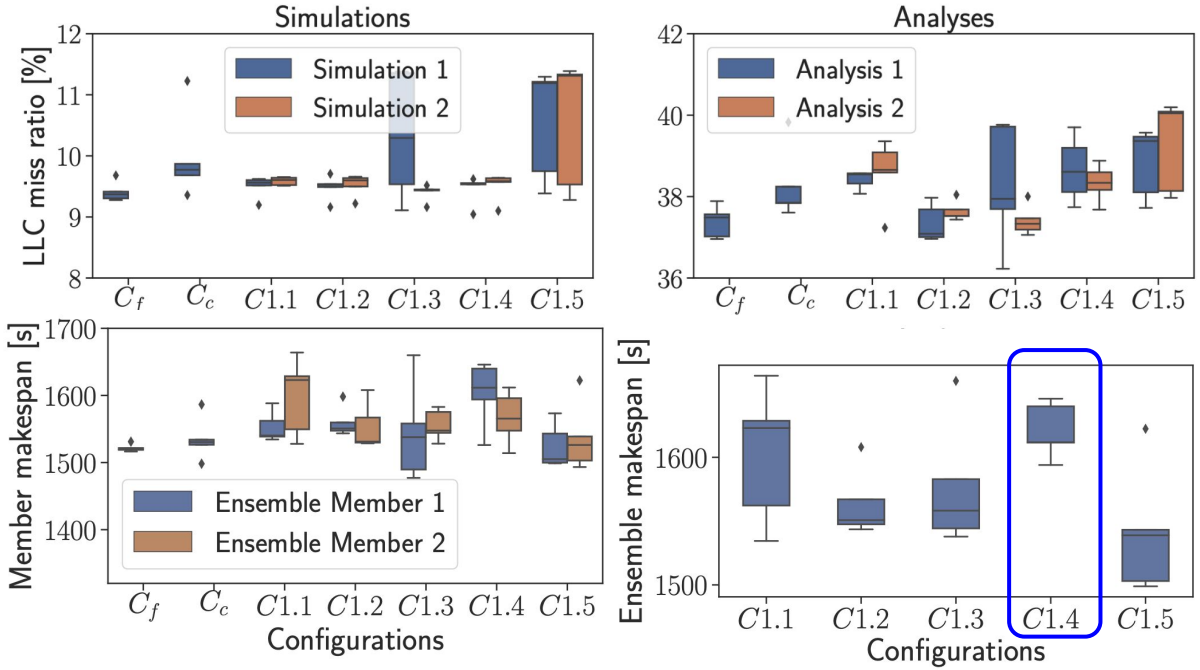
→ Minimize idle time

→ Minimize makespan

Characterization challenges

- Evaluating each ensemble component/member exclusively is hard to:
  - Generate a full picture of the workflow ensemble performance
  - Compare between different executions/configurations

- Without taking into account resources, the performance could be misleading



Config-uration	Number of computing nodes	Number of ensemble members	Node indexes			
			Ensemble member 1		Ensemble member 2	
			Simulation 1	Analysis 1	Simulation 2	Analysis 2
C_f	2	1	n_0	n_1	-	-
C_c	1	1	n_0	n_0	-	-
C1.1	3	2	n_0	n_2	n_1	n_2
C1.2	3	2	n_0	n_1	n_0	n_2
C1.3	3	2	n_0	n_0	n_1	n_2
C1.4	2	2	n_0	n_1	n_0	n_1
C1.5	2	2	n_0	n_0	n_1	n_1

C1.4 & C1.5 use fewer nodes than other configurations

Component placement

- The simulation is co-located with the analysis, iff $|s| = |s \cup a|$
- The simulation and analysis are assigned to different nodes, iff $|s| < |s \cup a|$

Set of node indexes where a simulation is executed

$$0 < \frac{|s|}{|s \cup a|} \leq 1$$

Set of node indexes where the coupled analysis is executed

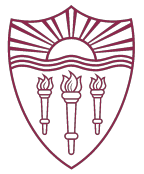
Placement indicator of ensemble member i with K_i analyses

$$CP_i = \frac{1}{K_i} \sum_{j=1}^{K_i} \frac{|s_i|}{|s_i \cup a_i^j|}$$

Mean of ratios forming by all (simulation, analysis) pairs

Maximize placement indicator prioritizes placements that minimize the number of computing resources (number of compute nodes) used by that ensemble member.

Performance indicator P_i of ensemble member i



1st stage — Resource usage (U)

Efficiency of single core usage

$$P_i^{\mathrm{U}} = \frac{E_i}{c_i}$$

c_i: Total number of cores used by ensemble member i

2nd stage — Resource allocation (A)

Efficiency of allocating ensemble components

$$P_i^{\mathrm{U,A}} = P_i^{\mathrm{U}} \times CP_i$$

CP_i: Placement indicator

3rd stage — Resource Provisioning (P)

Minimizing resources provisioned

$$P_i^{\mathrm{U,A,P}} = \frac{P_i^{\mathrm{U,A}}}{M}$$

M: Total number of compute nodes used by all ensemble members

Resource allocation (A) and resource provisioning (P) can be used interchangeably

Synthesis of performance indicators

- P_i can be either $P_i^{\mathrm{U}}, P_i^{\mathrm{U,A}}, P_i^{\mathrm{U,P}}, P_i^{\mathrm{U,A,P}} (= P_i^{\mathrm{U,P,A}})$

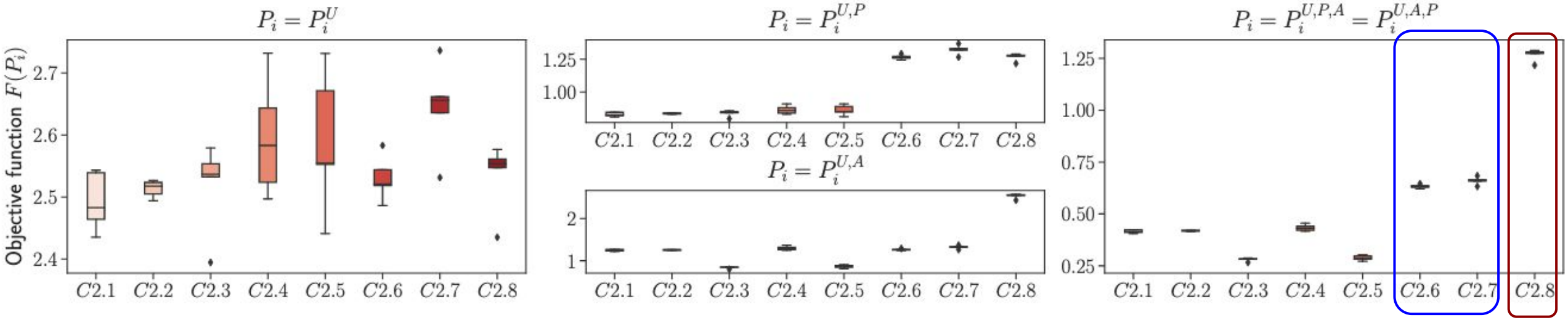
- The objective function of N ensemble members (the higher the better)

Maximize $F(P_i) = \overline{P} - \sqrt{\frac{1}{N}\sum_{i=1}^{N}(P_i - \overline{P})^2}$ where $\overline{P} = \frac{1}{N}\sum_{i=1}^{N} P_i$

Maximize average performance of ensemble members ← Mean

Standard deviation → Minimize variability among ensemble members

Two analyses per simulation



Configuration	Number of computing nodes (N)	Number of ensemble members	Node indexes					
			Ensemble member 1			Ensemble member 2		
			Simulation 1	Analysis 1.1	Analysis 1.2	Simulation 2	Analysis 2.1	Analysis 2.2
C2.1	3	2	n_0	n_2	n_2	n_1	n_2	n_2
C2.2	3	2	n_0	n_1	n_1	n_0	n_2	n_2
C2.3	3	2	n_0	n_1	n_2	n_0	n_1	n_2
C2.4	3	2	n_0	n_0	n_2	n_1	n_1	n_2
C2.5	3	2	n_0	n_1	n_2	n_1	n_0	n_2
C2.6	2	2	n_0	n_1	n_1	n_0	n_1	n_1
C2.7	2	2	n_0	n_0	n_1	n_1	n_0	n_1
C2.8	2	2	n_0	n_0	n_0	n_1	n_1	n_1

★
★★