The Power of Many: The Next Frontier

Shantenu Jha
Rutgers University and Brookhaven National Lab.

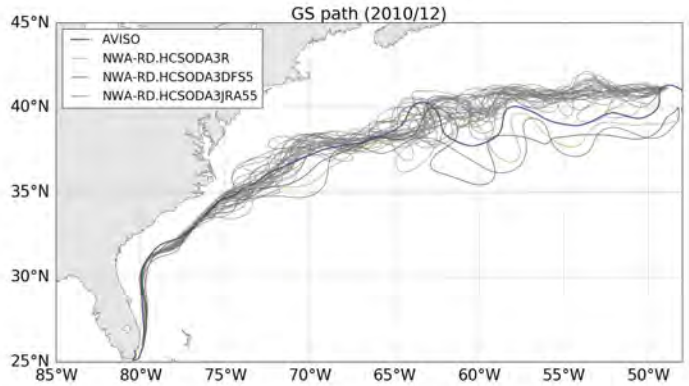
http://radical.rutgers.edu

Outline

- Ensemble Computational Model
  - Challenges of Ensemble Computational Model
- Executing Ensembles at Scale
  - **Performance Challenges:** Dynamic Resource Management
  - **Software Challenges:** Extensibility and Middleware Building Blocks
- Adaptive Ensemble Applications
  - ExTASY and Examples
- Next Frontier:
  - Drug Resistance using Adaptive Binding Free Energy calculations
  - **Learning Everywhere**! Using ML + HPC to enhance “Effective Performance”

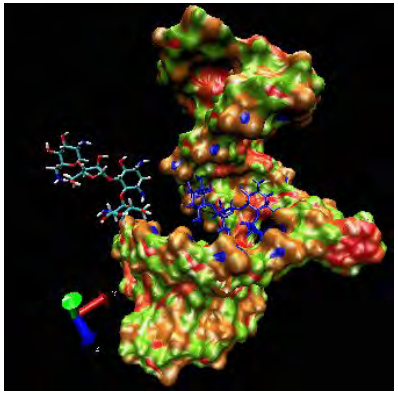
Ensemble Computational Model

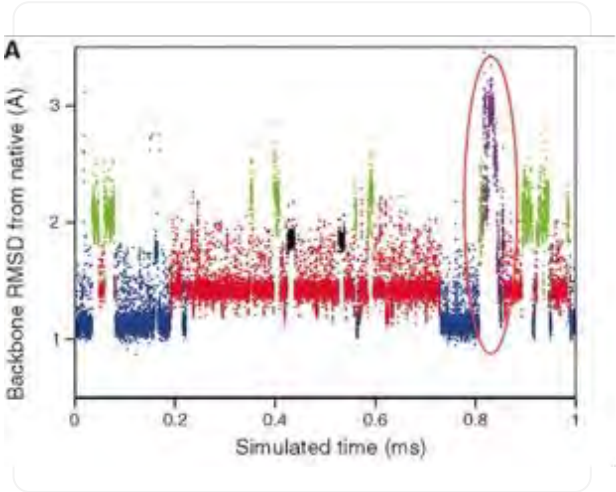
- Many applications formulated as **multiple** tasks, as opposed to large but **single** task.
- When the collective outcome of a set of tasks is important, defined as ensemble:
  - Distinct from HTC, typically tasks are I^4: (Independent, Idempotent, Identical, Insensitive to order)
- Performance is mix of HPC and HTC
  - Challenges go beyond traditional strong and weak scaling
  - Concurrent $N_E(t)$, total N_E, communication frequency ..
- Complexity of dependence resolution typically less than workflows



Ensemble Biomolecular Simulations

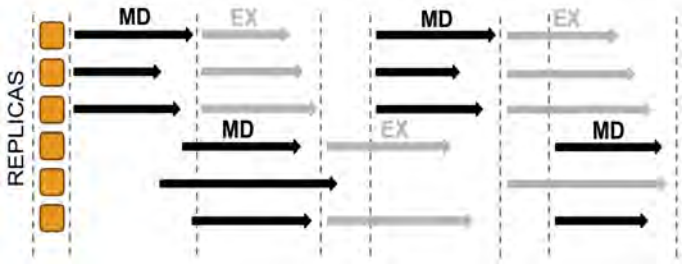
- Molecular Dynamics (MD): Newtons' Laws to integrate atoms over many timesteps
  - Immense success! (Chem, Nobel 2013)
- Single MD simulations not sufficient
  - Time scale vs quantitative accuracy
- Generate ensemble of simulations in parallel as opposed to one realization of process
  - Statistical approach: **O(10^6 - 10^8)** !
- Specialized hardware, e.g., DE Shaw "Anton" valuable, *but can ensemble-based algorithms do better than specialized hardware*?

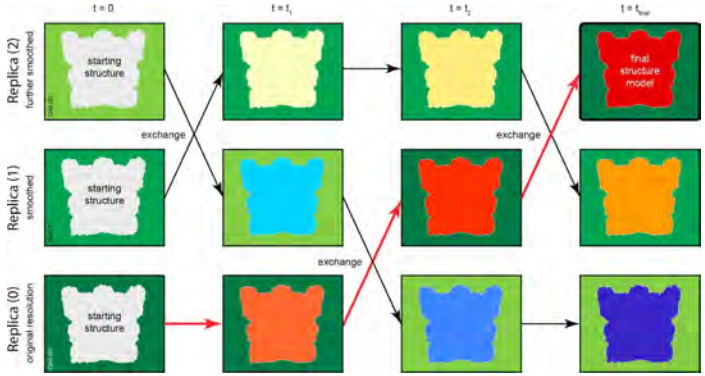




Adaptive Ensemble Algorithms: Variation on a theme

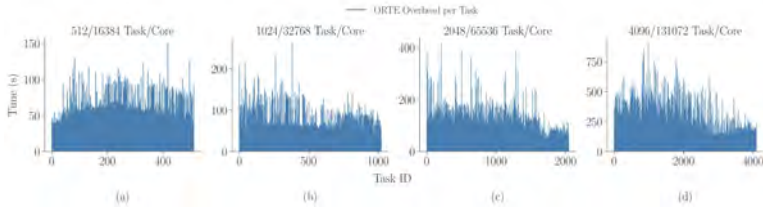
- Ensemble-based methods necessary, but not sufficient !
- **Adaptive** Ensemble-based Algorithms: Intermediate data, determines next stages
- Adaptivity: How and What
  - Internal data used: Simulation generated data used to determine “optimal” adaptation
  - External data used, e.g., experimental or separate computational process.
  - What: Task parameter(s), order, count, ….

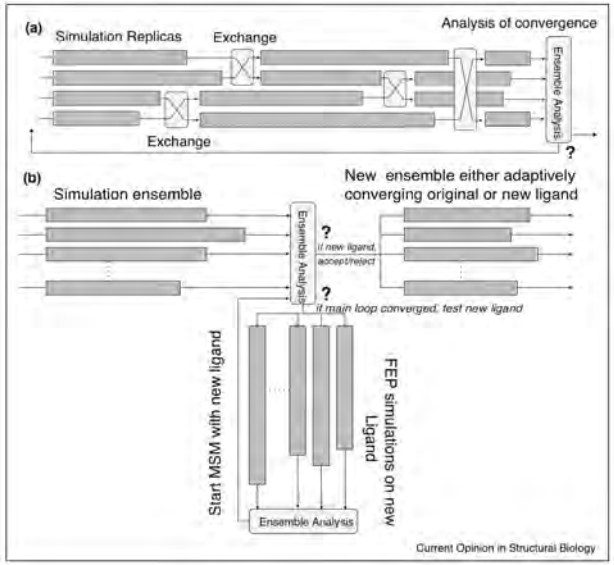




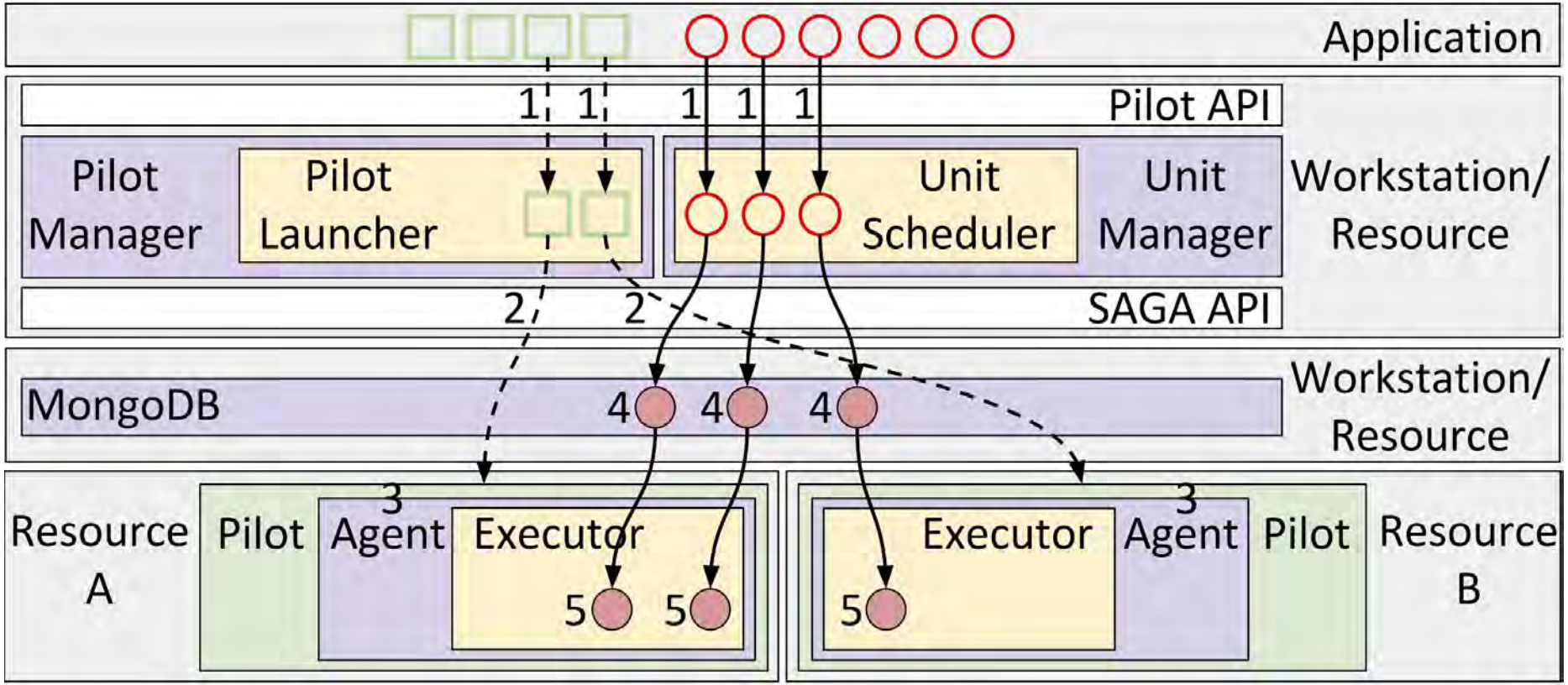
Ensemble Simulations at Scale: Challenges

- **Resource Management** for $O(10^{5-6})$ tasks -- each is independent executing program!
  - Exascale ~$O(10^{6-9})$
- **Application requirements and resource performance must be dynamic**
  - Abstraction of static perf. is inadequate!
  - Implications on perf. portability & scaling
- **Execution Model** of heterogeneous tasks on heterogeneous and dynamic resources.
- System software that support encoding algorithms that express adaptivity, even statistically ("approximately")?
  - Managing interactions (coupling) between tasks
  - **…..**



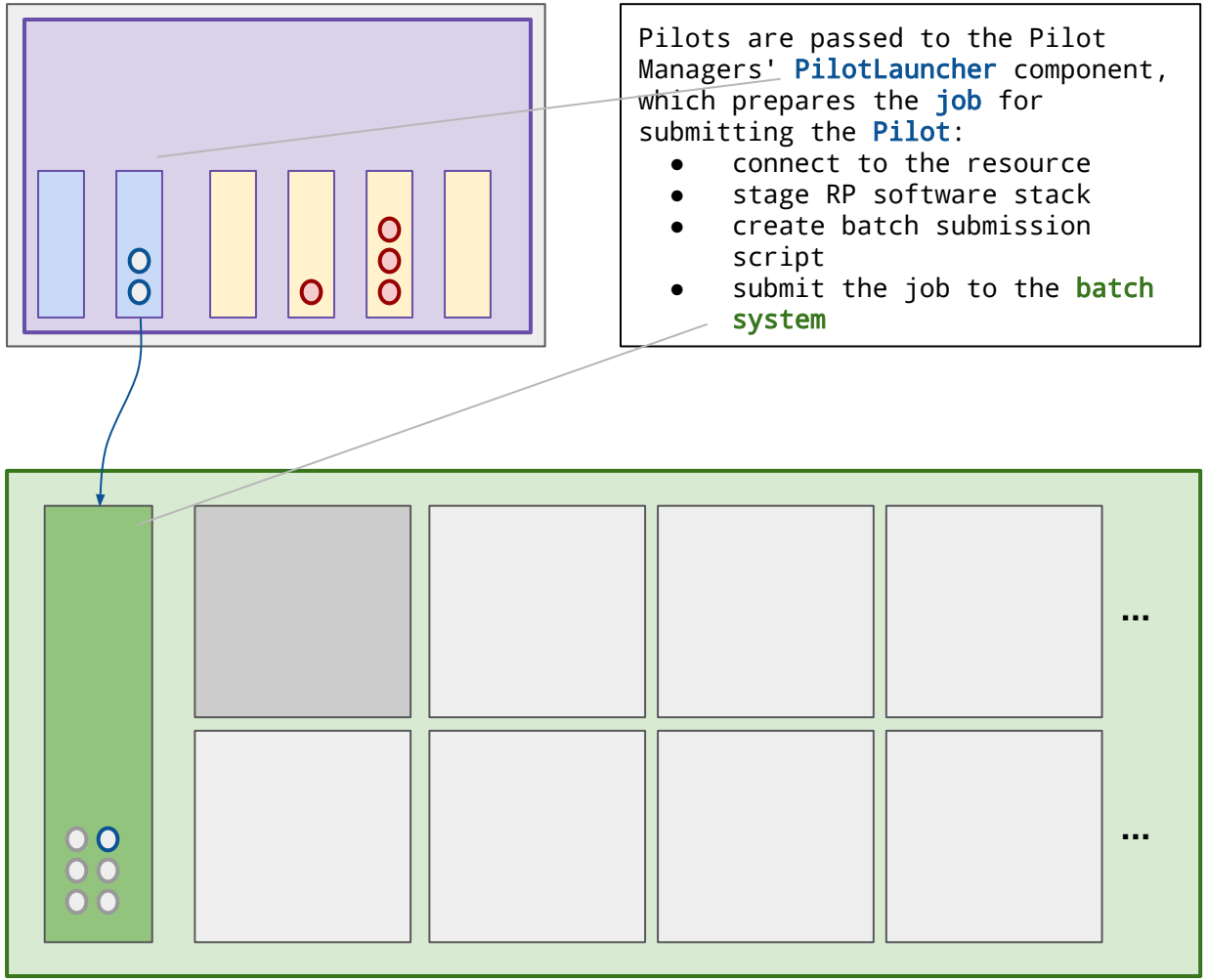


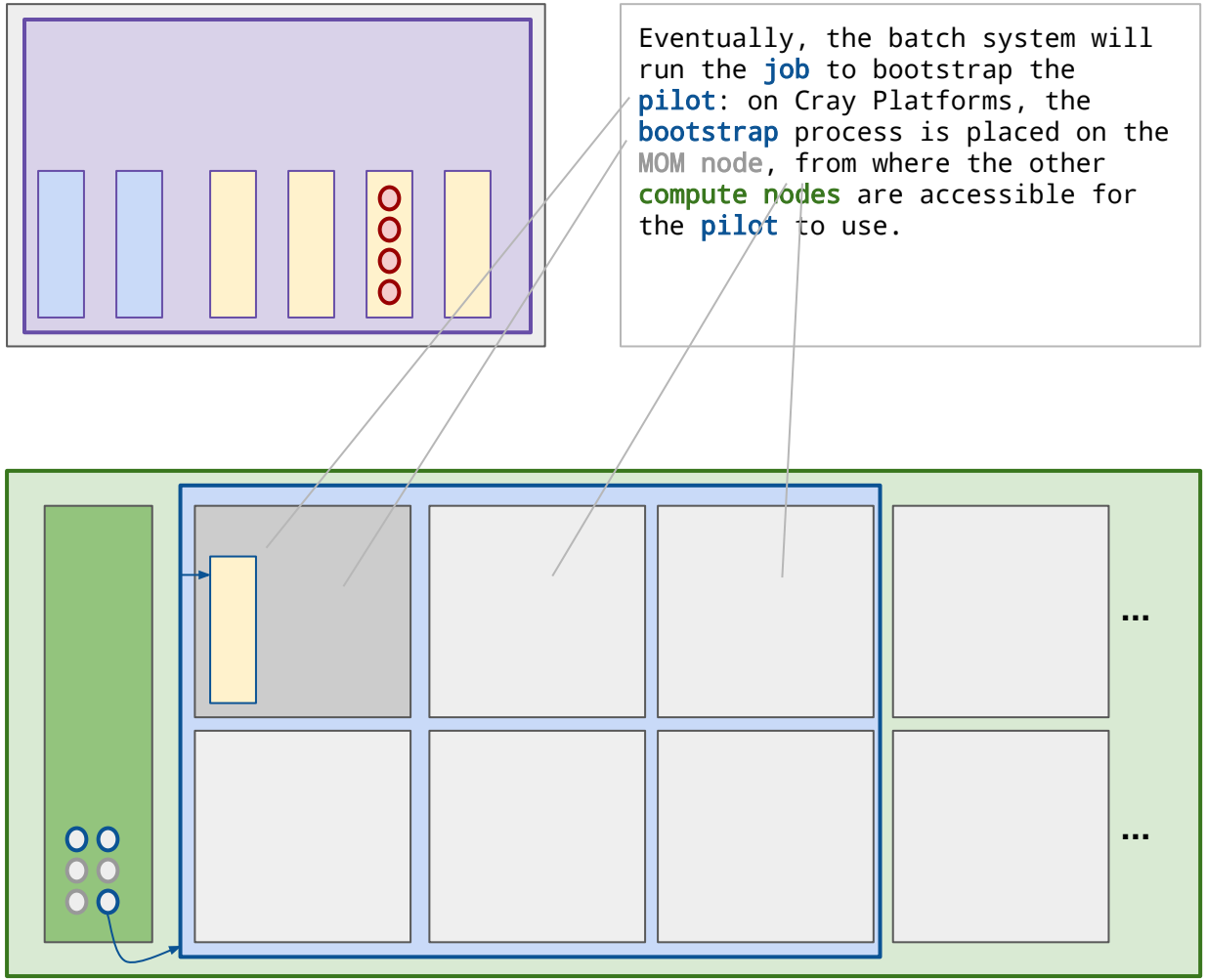
RADICAL-Pilot: Execution Model

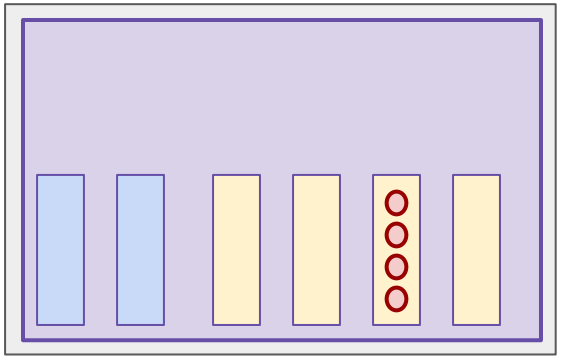


Pilot-Abstraction: Summary

- Run multiple tasks **concurrently** and **consecutively** in a **SINGLE** batch job:
  - Tasks are programs, i.e., executables, not methods, functions, threads
  - Tasks are executed within the scope of the batch job
- Late binding:
  - Tasks are NOT packaged into the batch job before submission.
  - Tasks are scheduled and then placed within the batch job at runtime.
- Task and resource heterogeneity:
  - Scheduling, placing and running CPU/GPU/OpenMM/MPI tasks in same batch job
  - Use single/multiple CPU/GPU for the same tasks and/or across multiple tasks.

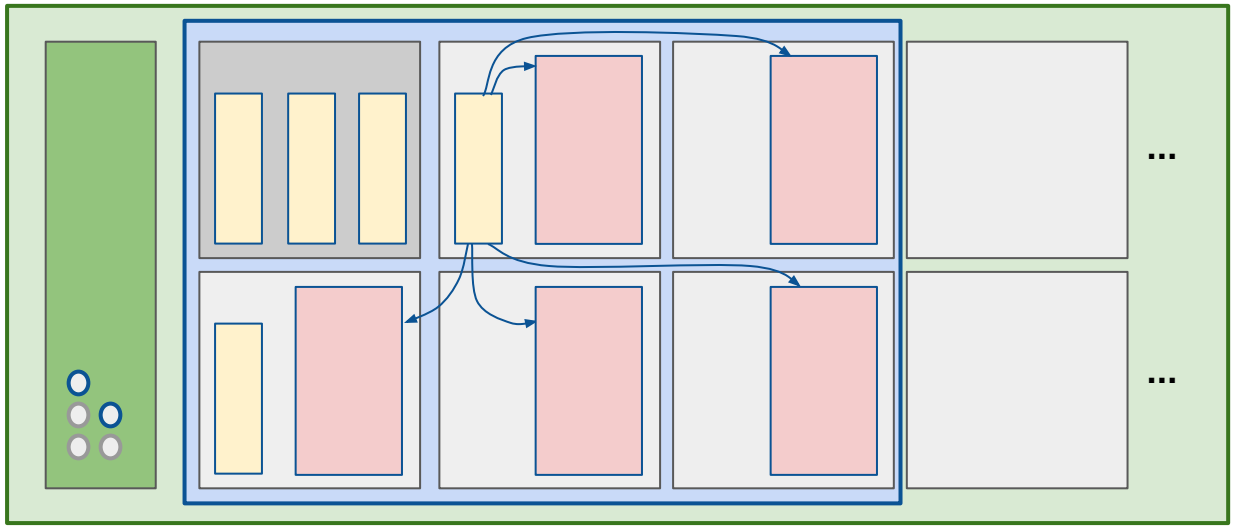

Pilots are passed to the Pilot Managers' PilotLauncher component, which prepares the job for submitting the Pilot:
connect to the resource
stage RP software stack
create batch submission script
submit the job to the batch system
...
...

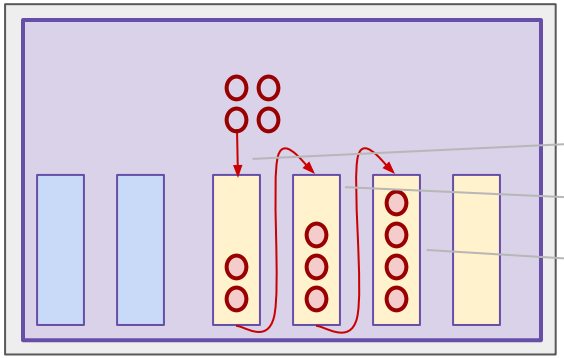

Eventually, the batch system will run the job to bootstrap the pilot: on Cray Platforms, the bootstrap process is placed on the MOM node, from where the other compute nodes are accessible for the pilot to use.
...
...



The first unit executor component will create an OpenMPI Distributed Virtual Machine (**ORTE DVM**) across all compute nodes.

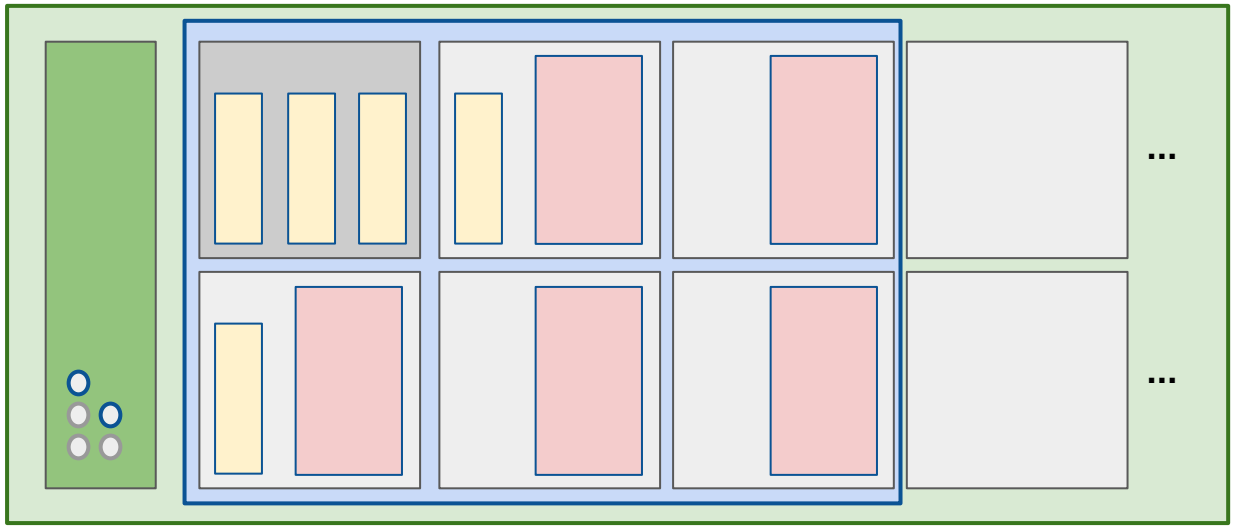
At this point, the pilot is ready to receive and execute compute units.

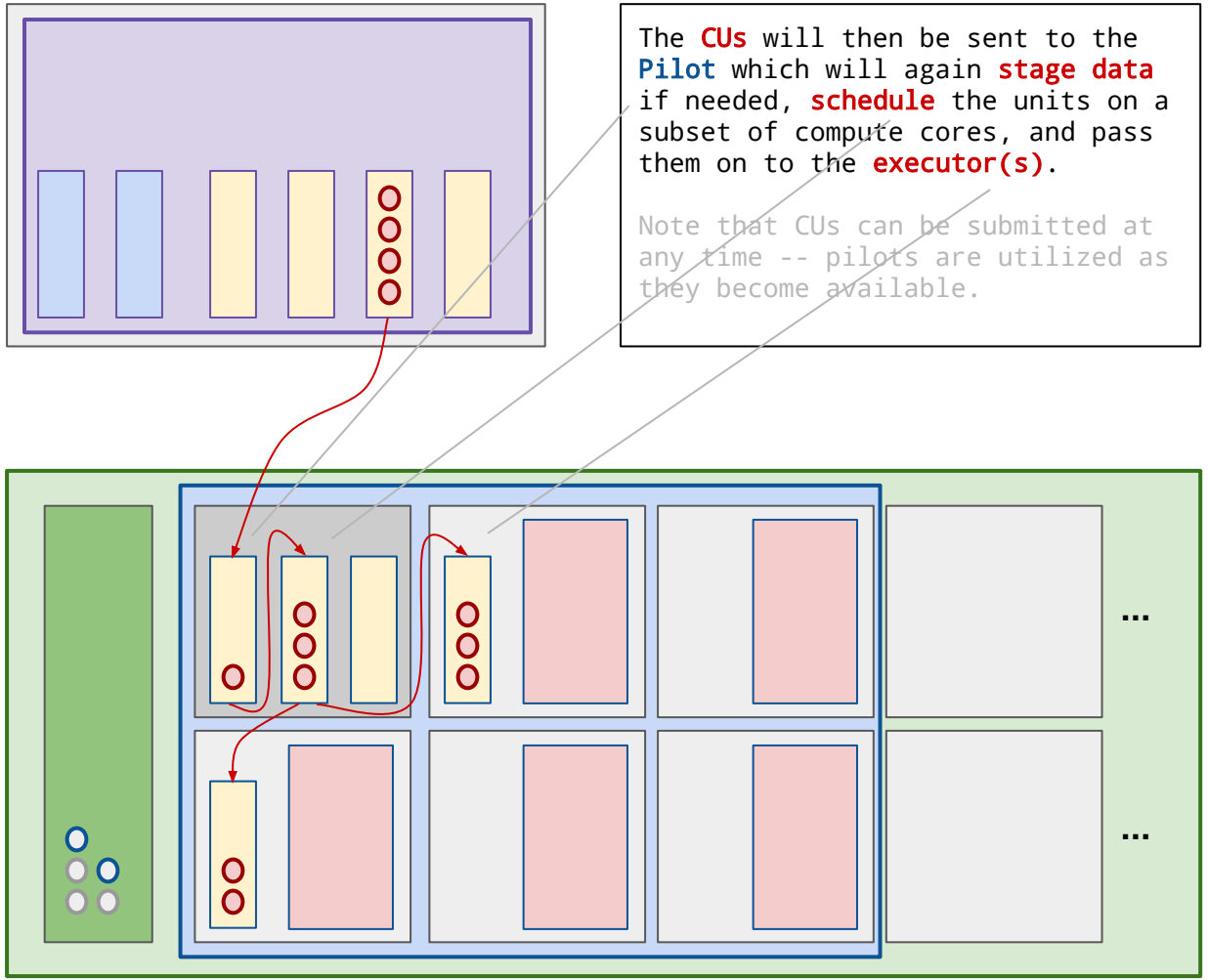


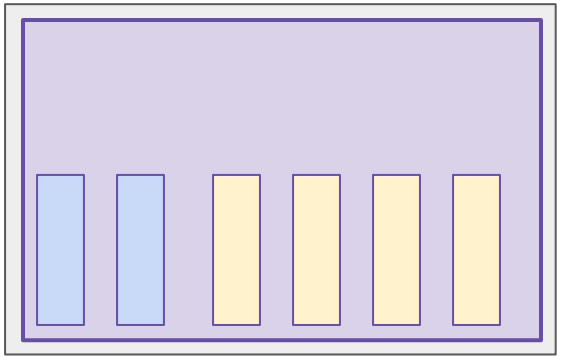


We will now look into the unit execution path:

When a **UnitManager** receives *new* requests to execute **CUs**, its **scheduler** will assign them to an available **Pilot,** and the **input stager** will transfer the CU's input data.



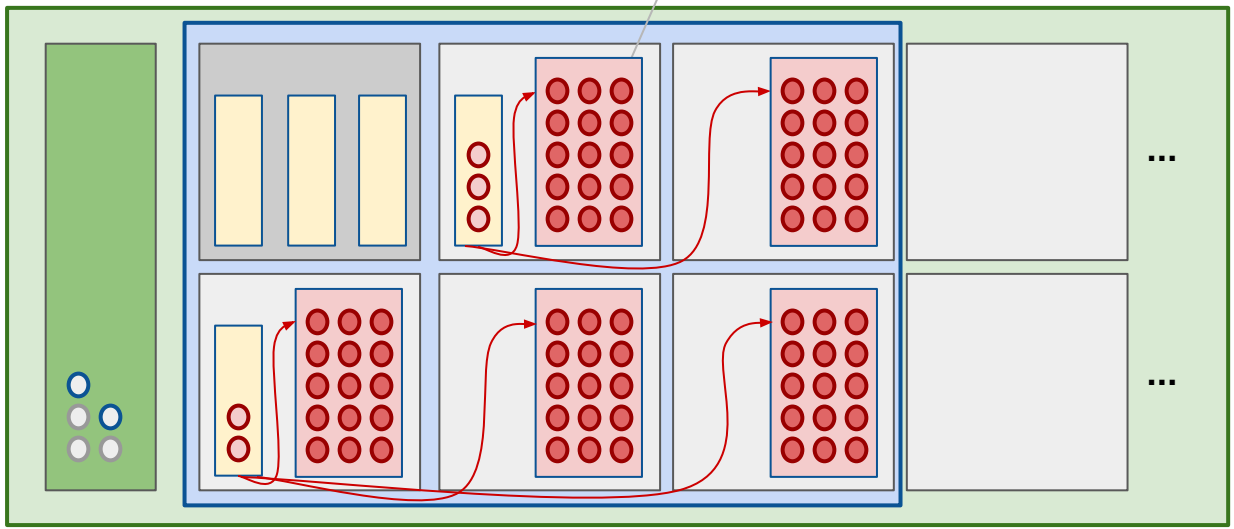

The CUs will then be sent to the Pilot which will again stage data if needed, schedule the units on a subset of compute cores, and pass them on to the executor(s).
Note that CUs can be submitted at any time -- pilots are utilized as they become available.
...
...

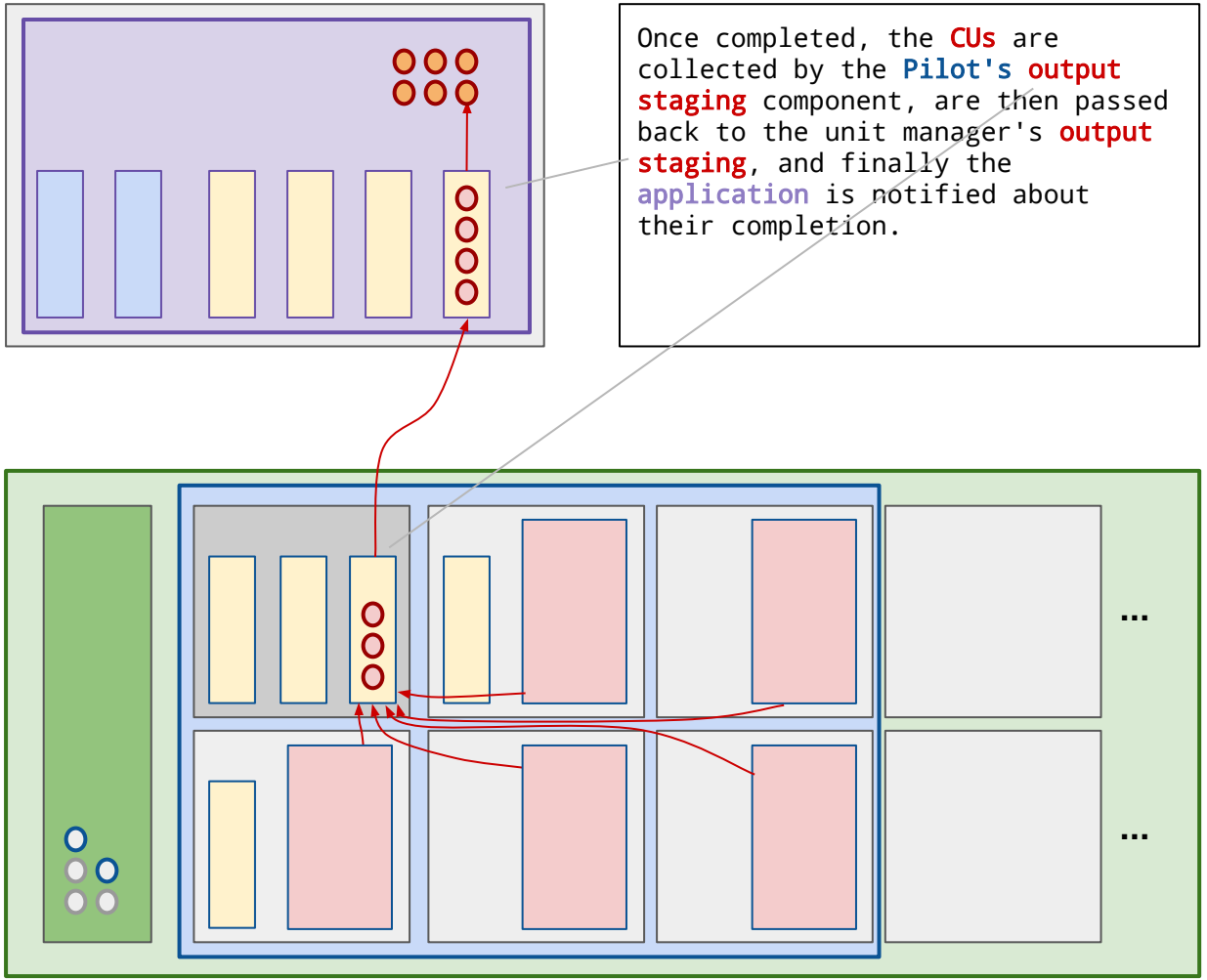


The executors will pass the CUs on to the **ORTE DVM** for execution.

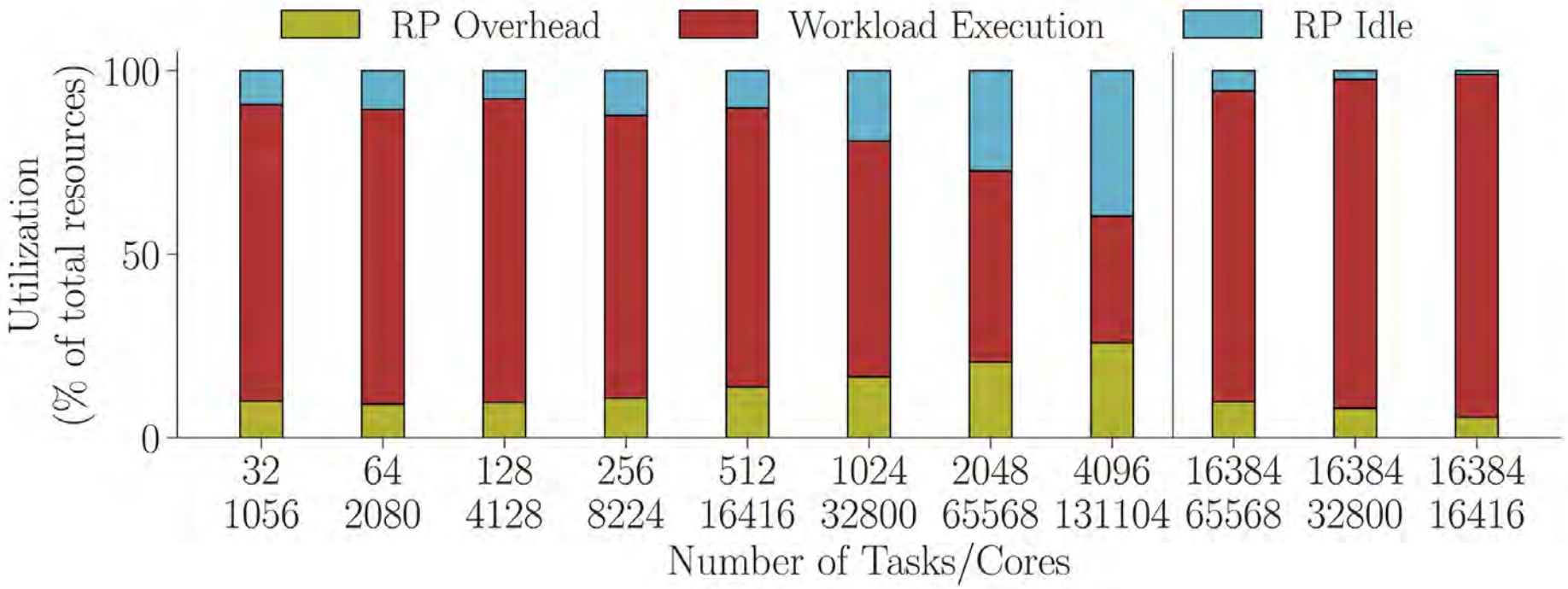
The executor's performance and the DVM are optimized for high throughput, ensuring high system utilization.

RP can mix MPI / non-MPI jobs, GPU support is coming soon.



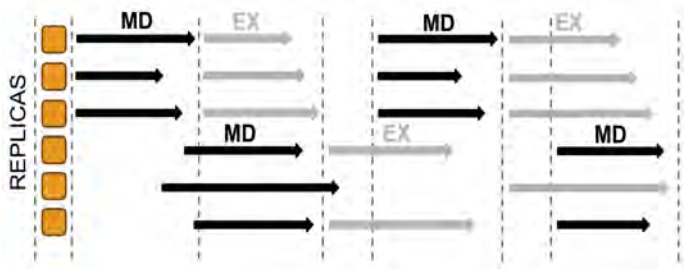

Once completed, the CUs are collected by the Pilot's output staging component, are then passed back to the unit manager's output staging, and finally the application is notified about their completion.
...
...

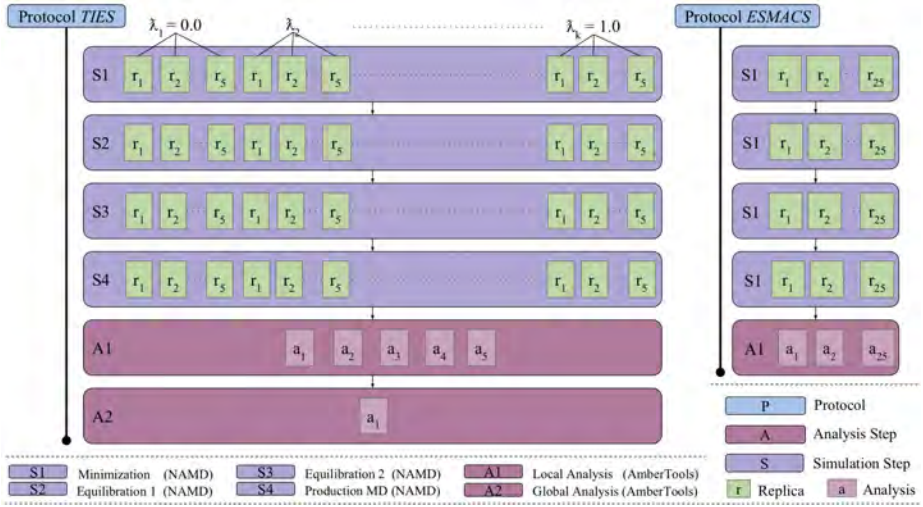
RADICAL-Pilot: Resource Utilization Performance

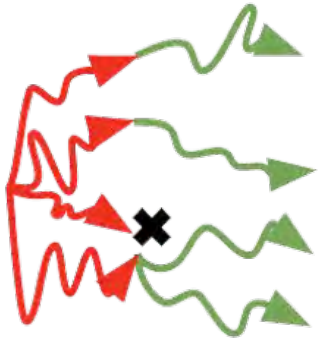


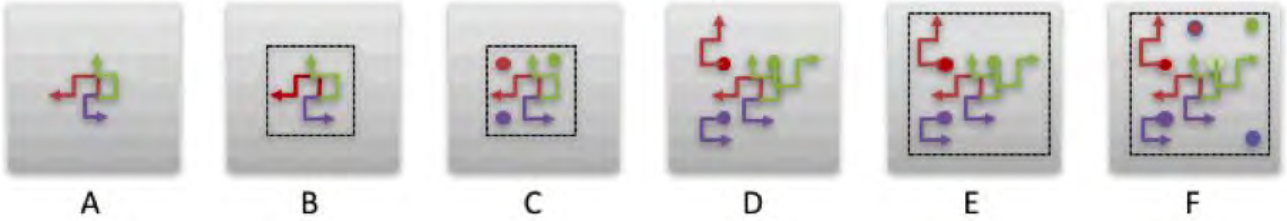
Adaptive Ensemble Algorithms: Variation on a theme

Better, Faster, Greater sampling

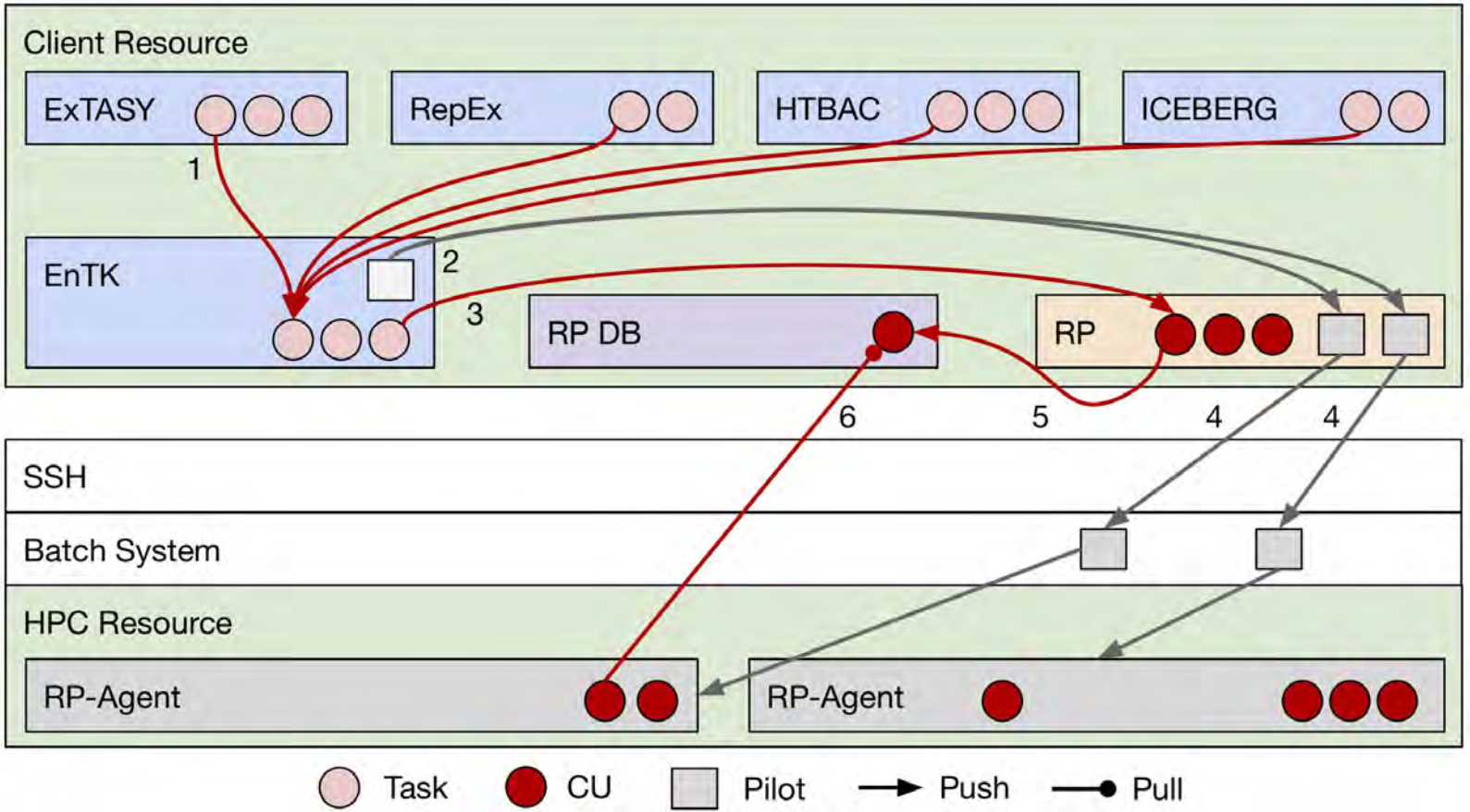






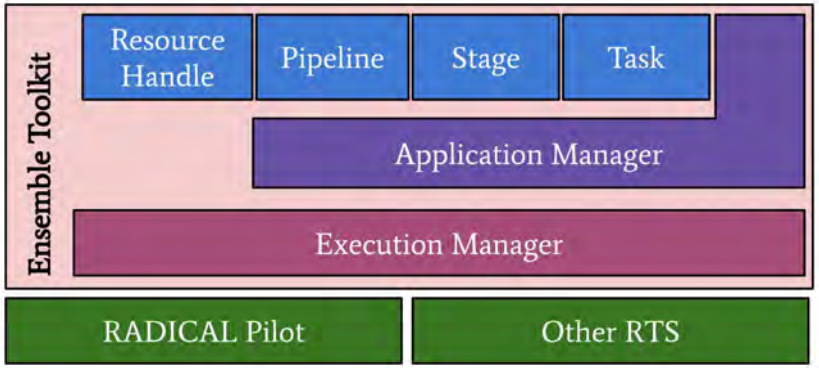


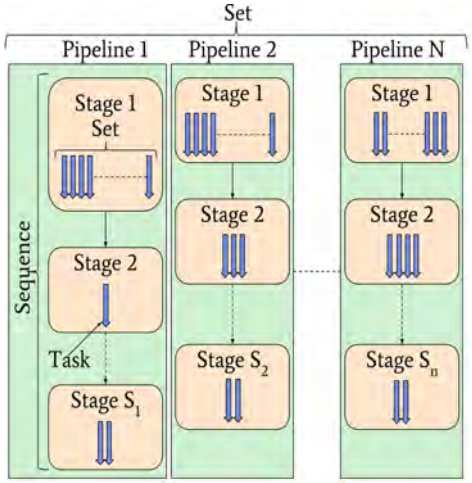
EnTK: Supporting Several Domain Specific Workflows



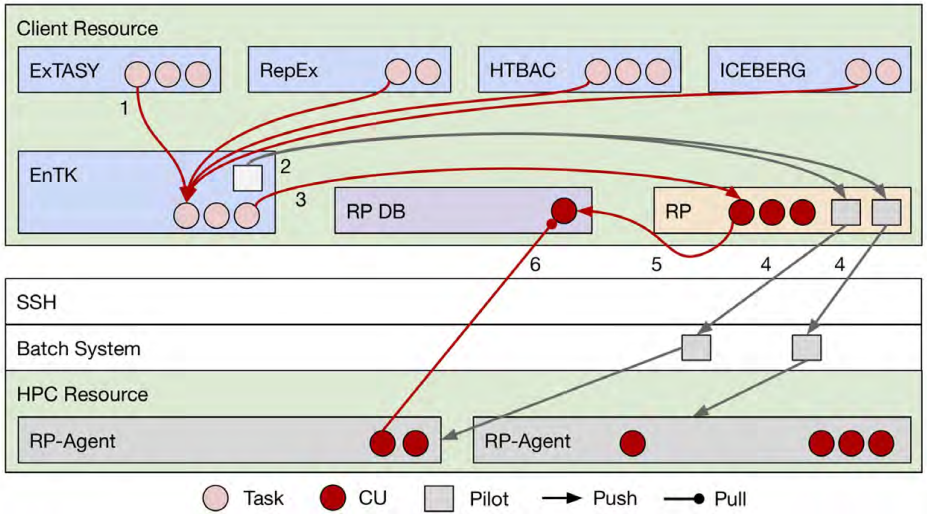
EnTK: Building Block for Ensemble based Applications

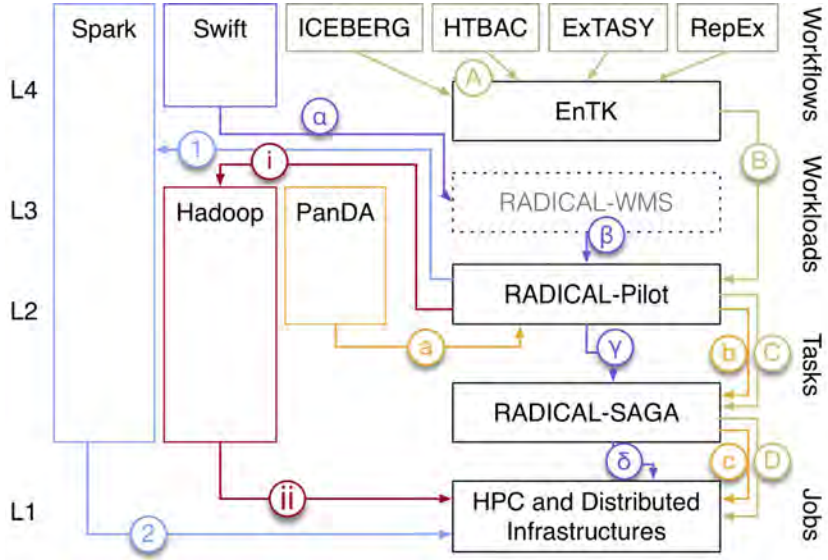
- **Ensemble-Toolkit (EnTK):** Promoto ensembles as a first-class programming and execution entity.
  - (i) Facilitate expression of ensemble based applications, (ii) manage complexity of resource acquisition, and (iii) task execution.
- **Architecture:**
  - User facing components (blue); Workflow management components (purple); Workload management components (red) via runtime system (green)
- **PST Programming Model:**
  - **Task:** an abstraction of a computational process and associated execution information
  - **Stage:** a set of tasks without dependencies, which can be executed concurrently
  - **Pipelines:** a list of stages, where stage “i” can be executed after stage “i−1”





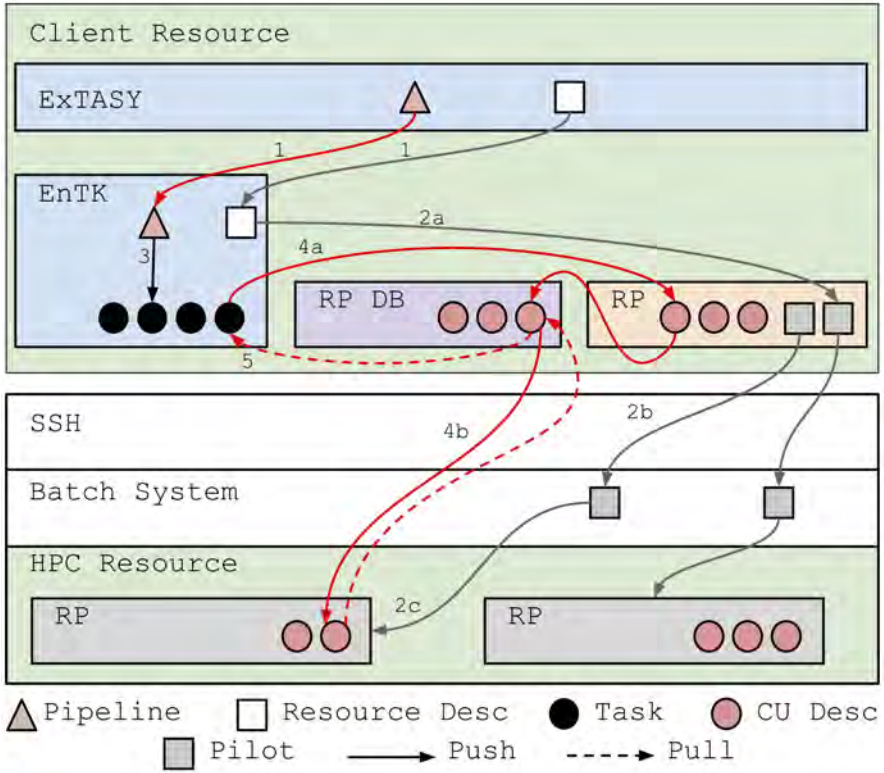
Software Systems Challenge: Specificity with Performance

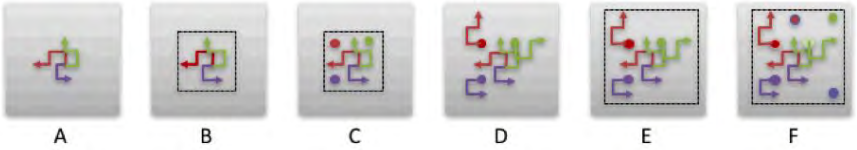




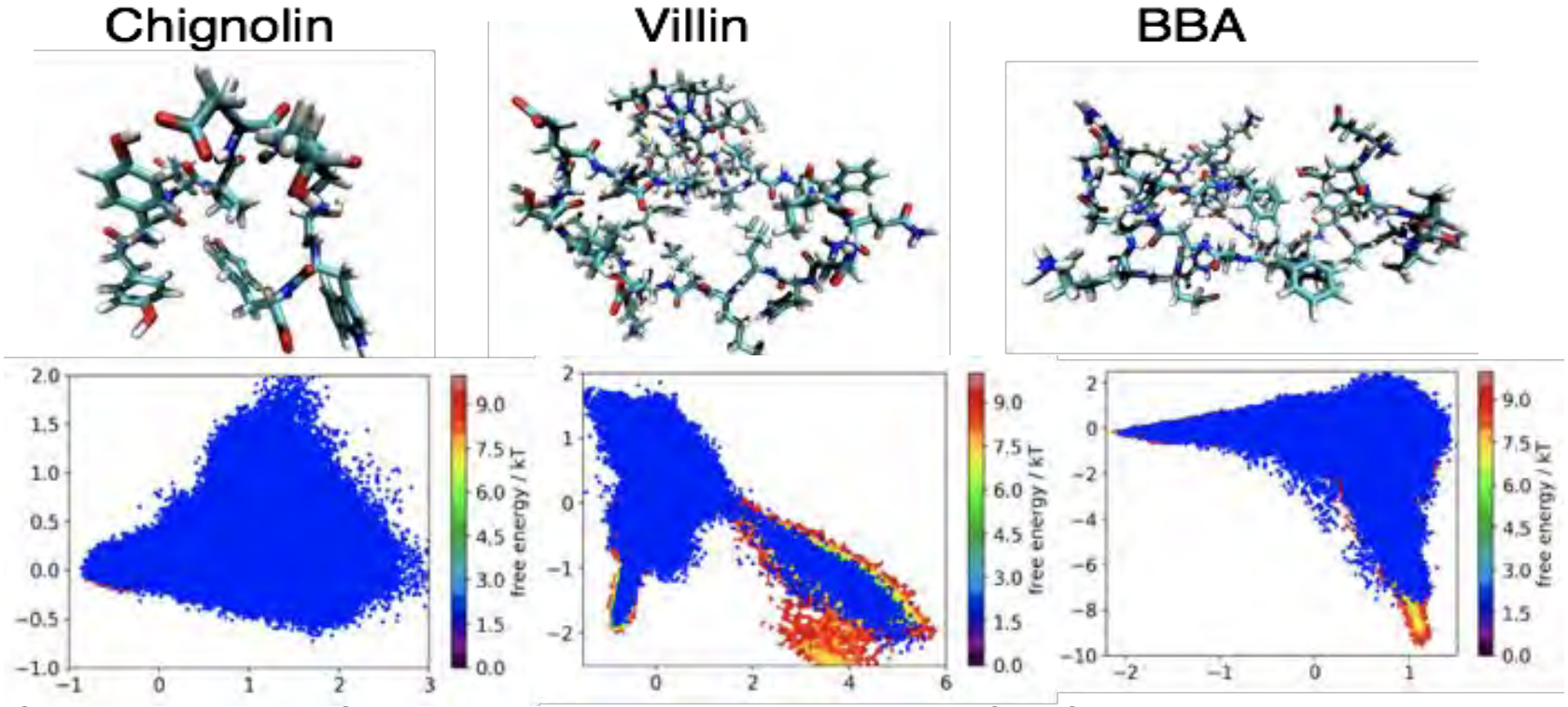
Middleware Building Blocks for Workflow Systems https://arxiv.org/abs/1903.10057

ExTASY: Domain Specific Workflow System



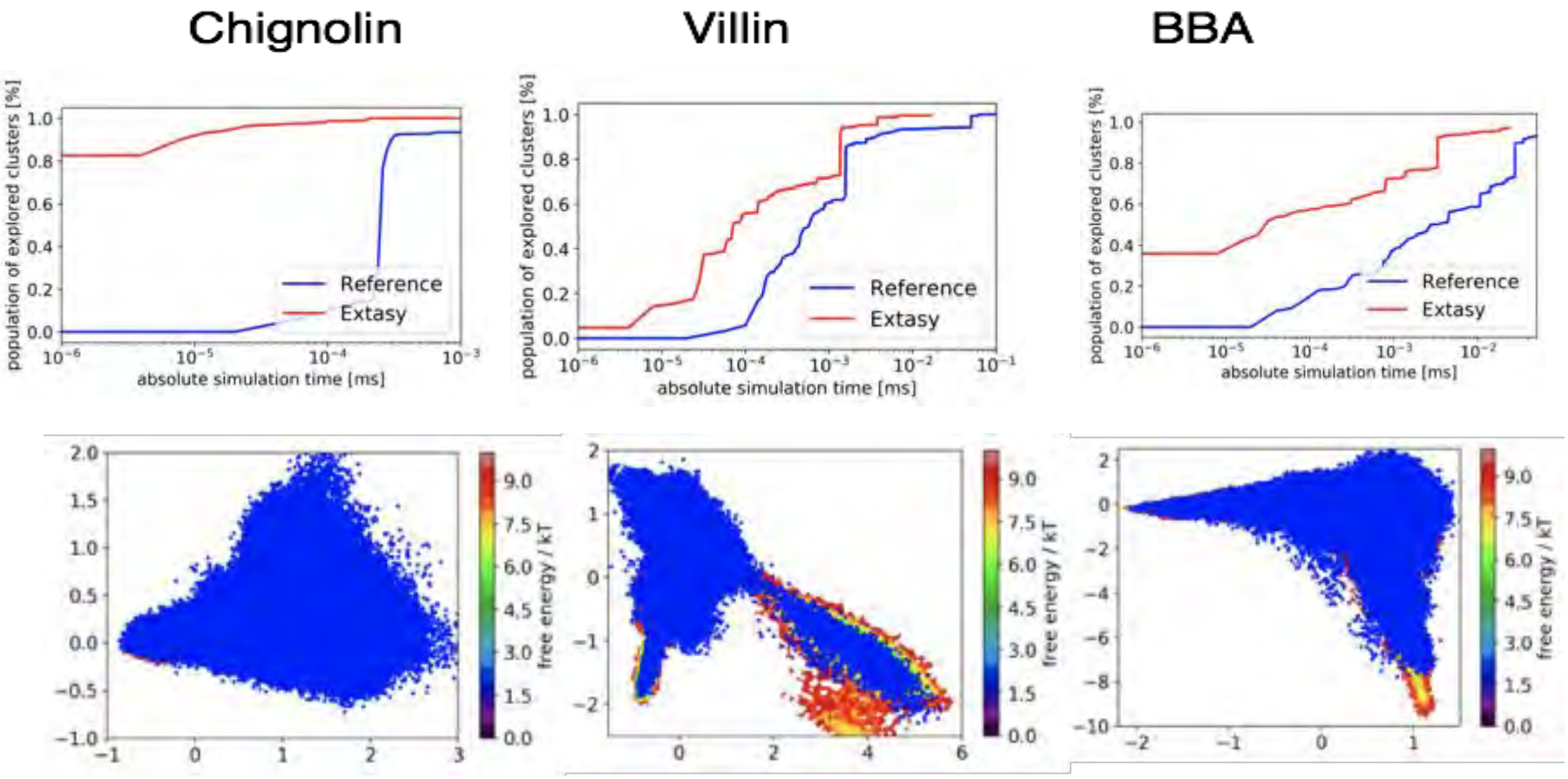


ExTASY: Enhanced Conformational Sampling



- Comparing Adaptive Sampling results with [2] Full exploration of the free energy landscape
- [2] K. Lindorff-Larson, S. Piana, R. O. Dror, and D.E. Shaw, Science 344, 517 (2011)

Adaptive Ensemble MD vs Conventional MD

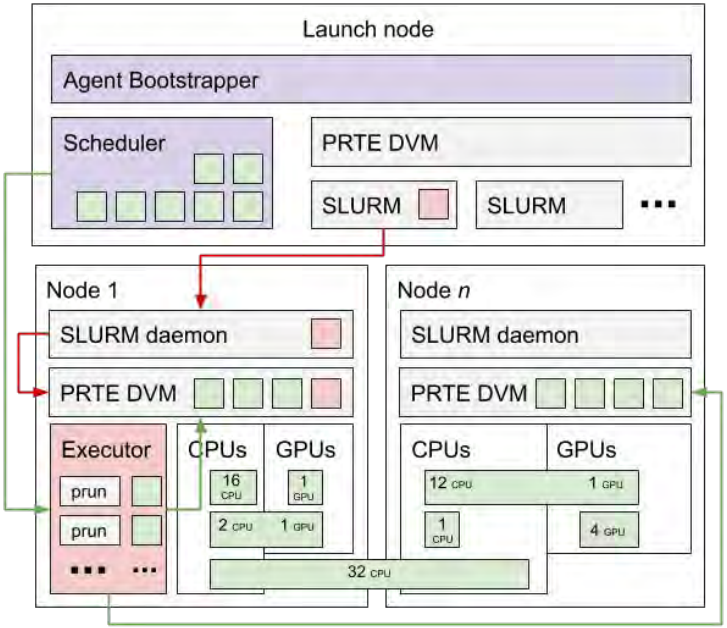


Outline

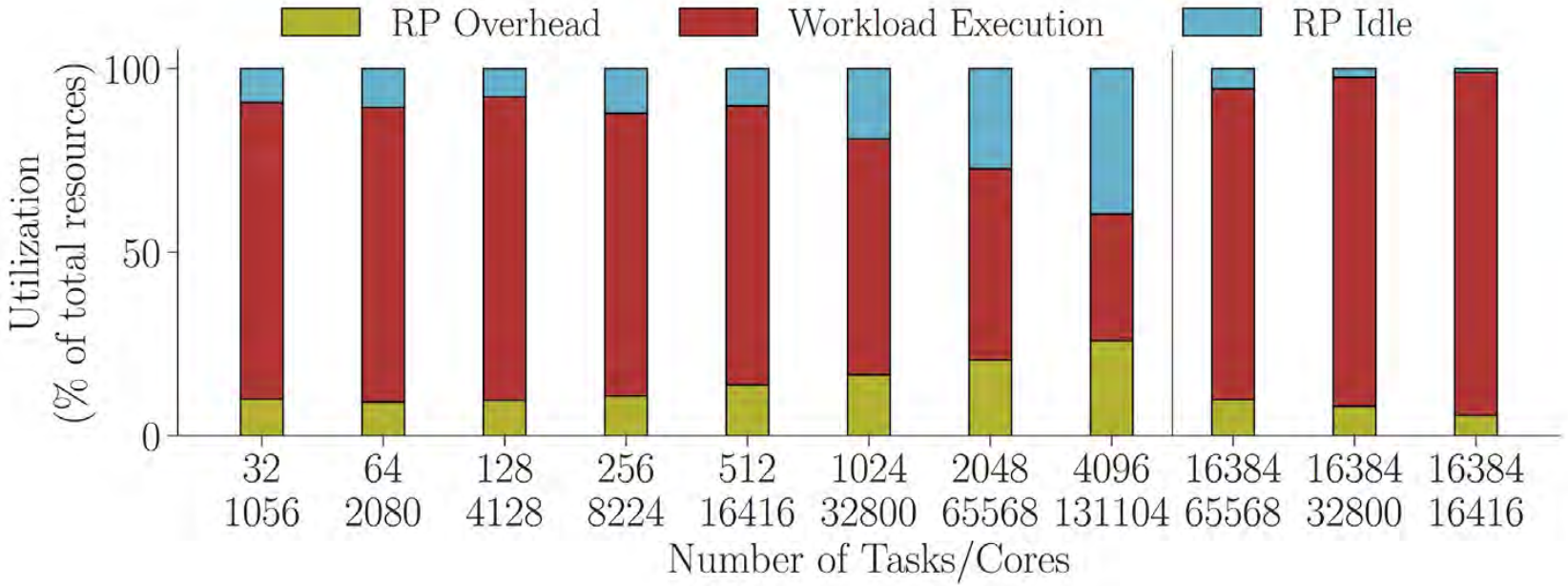
- Ensemble Computational Model
  - Challenges of Ensemble Computational Model
- Executing Ensembles at Scale
  - **Performance Challenges:** Pilot-Abstraction and RADICAL-Pilot
  - **Software Challenges:** Middleware Building Blocks and Ensemble Toolkit
- Adaptive Ensemble Applications: Examples
  - Adaptive Ensemble versus “conventional” MD simulation
  - Adaptive Ensemble versus “vanilla” Ensemble MD simulation
- **Power of Many: The Next Frontier:**
  - Next generation leadership platforms
  - **Learning Everywhere**! Using ML + HPC to enhance “Effective Performance”

RADICAL-Pilot on Leadership Class Machine

- Can we get performance agnostic of batch queue systems and MPI flavour?
  - LSF, PBS, SLURM, … ?
  - MVAPICH, … MPI flavours?
- **PMI-X: P**rocess **M**anagement **I**nterface for E**X**ascale https://github.com/pmix/pmix/wiki
  - **PRRTE: P**MI-X **R**eference **R**un**T**ime **E**nvironment https://github.com/pmix/prrte
- PMI used by MPI implementations, batch system
- Private DVM, concurrent tasks
- Pros: heterogeneous tasks (as with JSRUN), (potentially) fast, **portable**
- Cons: Young code; emerging official support



RADICAL-Pilot: Resource Utilization Performance (Titan)



"... The PMIx community has committed to reducing or eliminating the time spent in these stages to achieve an overall goal of launching and connecting exascale applications in under 30 seconds. For purposes of tracking this goal, the community has adopted its baseline test as being the time ***required to start an application and complete MPI Init, with all processes having all required information to communicate at that point, using an application size of 50K nodes supporting up to 1M individual processes****…"* ***Castain et al, Parallel Computing 2018***

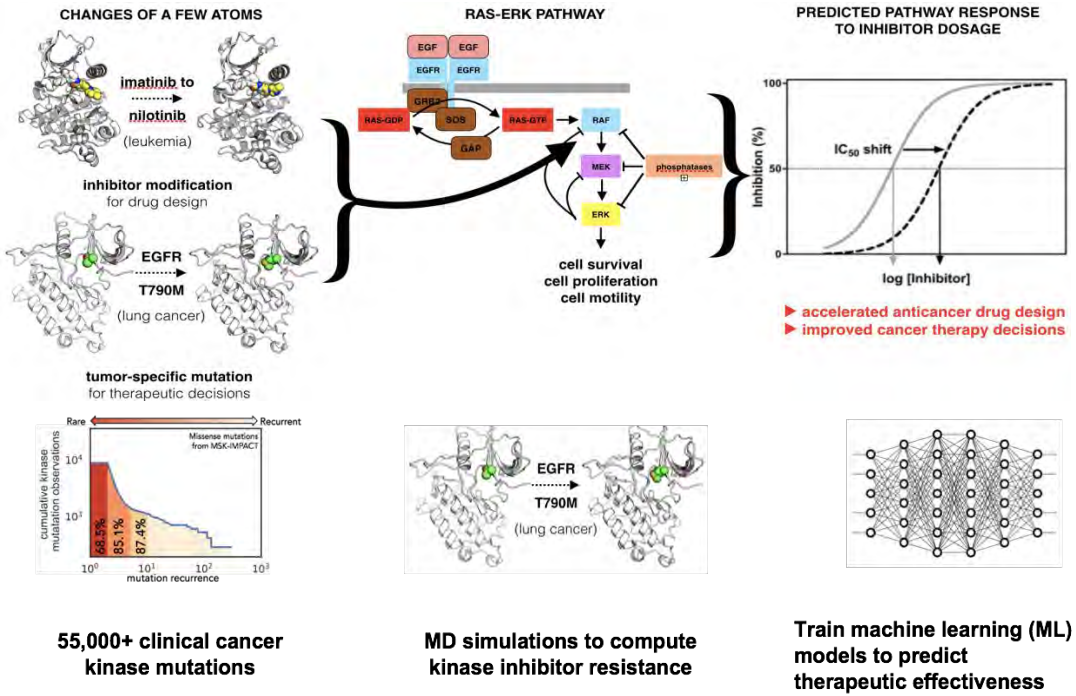
MLforHPC: Classification and Examples

MLforHPC: Using ML to enhance HPC applications and systems

- **MLAutoTuning:** Using ML to configure (autotune) ML or HPC simulations.
  - Nanoparticles Ionic distribution: ANN regression models
- **MLafterHPC:** ML analyzing results of HPC as in trajectory analysis and structure identification in biomolecular simulations
  - Using deep learning approaches for MD trajectory
- **MLaroundHPC:** Using ML to learn from simulations and produce learned surrogates for the simulations or parts of simulations.
  - Adaptive Sampling: Predicting next steps in MD
- **MLControl:** HPC simulations in control of experiments and/or objective driven computational campaigns. Simulation surrogates allow real-time predictions.
  - *Objective Driven Drug Candidate Selection*
- “Learning Everywhere: Learning for Effective HPC” https://arxiv.org/abs/1902.10810

INSPIRE: Integrated (ML-MD) **S**calable **P**rediction of **RE**sistance

- Chemical space of drug design in response to mutations very large. 10K -100K mutations; too large for HPC simulations alone!
- Develop methods that use: (i) simulations to train machine learning (ML) models to predict therapeutic effectiveness; (ii) use ML models to determine which drug candidates to simulate.



Early Science Project on NSF Frontera. DD Award on Summit.

A collaboration between BNL/Rutgers (Jha), Chicago (Stevens), Memorial Sloan Kettering (Chodera), UCL (Coveney)

Learning EveryWhere!

MLforHPC: Using ML to enhance HPC applications and systems

- **MLautotuning**, **MLafterHPC, MLaroundHPC, MLControl**
- Proposed a reference architecture for Learning Everywhere -- concept paper by Fox & Jha **arXiv:1902.10810** (IPDPS 19 Workshop)
- Reference Architecture: Adaptive (Heterogenous) Ensemble Applications comprised of:
  - N_L (L = Learning Element)
  - N_S (S = Simulation Element)
  - N_D (D = Data Source / Generation)
  - N_L / N_S / N_D are time dependent and so is the coupling between them.
- **Learning Everywhere resource management and system software challenges are similar to adaptive ensemble!**
  - Dynamic, adaptive, and heterogeneous workflows

Summary

- **Power of Many: Algorithmic and methodological advances are needed**
  - Adaptive execution of large ensembles
  - Adaptive algorithms impose new software challenges https://arxiv.org/abs/1804.04736
- **Importance of Abstractions and Middleware building blocks**
  - Support heterogeneous workloads and resources (i) ~O(10K) MPI simulations and tasks; (ii) Price for heterogeneity (generality); (iii) Ready for scheduling optimization; (iv) O(100K): Message subsystem will hit!
  - Ensemble Toolkit (F) & RADICAL-Pilot (P)
- **Learning Everywhere**
  - Learning systems translating the reference architecture to software systems

Thank You!

Acknowledgement: Blue Waters NSF #1713749 and sub-award to Rutgers via UIUC on NSF SAVI GECAT (NSF #1440733) *GECAT-Global Initiative to Enhance Collaborative Computing and Analysis Tools*