More Power to the Many: Scalable Ensemble-based Simulations and Data Analysis

Shantenu Jha
Brookhaven National Lab & Rutgers University.

http://radical.rutgers.edu
Why a “Fresh Perspective” to Workflows?

- Initially “Monolithic” Workflow systems with “end-to-end” capabilities
  - Workflow systems were developed to support “big science” projects.
  - Software infrastructure was “fragile”, unreliable, missing services

- Workflows aren’t what they used to be!
  - More pervasive, sophisticated but no longer confined to “big science”
  - Extend traditional focus from end-users to workflow system/tool developers!
  - Prevent vendor lock-in

- Building Blocks (BB) permit workflow tools and applications can be built
  - Diverse “design points”; unlikely “one size fits all”; last mile distinction
A Layered View of Distributed Cyberinfrastructure

- Propose four layers:
  - L4: Workflows [Application semantics]
  - L3: Workload execution and management (WLMS) [Workload]
  - L2: Task runtime system (TRS) [Tasks]
  - L1: Resource layer [Jobs]

- Workflow: Complete description of what and when needs to be executed.

- Workload: A set of related tasks and their execution descriptions.
  - Payload of the workflow: description of what needs to be executed, not how.
  - Malleable: can be “shaped”
- BB to support workflows, and the development of workflow tools
- A "laboratory" for testing ideas, support production science
- Stand alone, as well as vertical integration and horizontal extensibility
RADICAL-Cybertools: Building Blocks for Workflows

- A “laboratory” while supporting production grade workflows and workflow tools.
  - Consistent with HPC & scale
- Integrate with existing tools:
  - Swift, Fireworks, PanDA, Binding Affinity Calculator (BAC)
  - Distinct points of integration, vertical integration and horizontal extensibility
  - Need “faster” start, “scalable” (more tasks) and “better” (resource utilization)
- Novel tools and libraries:
  - ExTASY, RepEx, HTBAC, Seisflow,..
RCT BB: From Streaming to Seismic Data

- Design HPC stream processing systems
  - Resource contention limits scalability of reconstruction algorithms

- Supporting Seismic Physics Workflows

- Task Parallel Analysis for Trajectory Data
RADICAL-Pilot: Implementation of Pilot-Abstraction

- “.. a scheduling overlay which generalizes the recurring concept of utilizing a placeholder as a container for compute tasks”

- Decouples workload from resource management

- Enables the fine-grained spatio-temporal control of resources

- Build higher-level frameworks without explicit resource management

- Provides building block for late-binding of workloads on HPC

RADICAL-Pilot: Resource Utilization Performance

![Bar chart showing resource utilization performance for different numbers of tasks/cores. The chart displays the percentage of resources utilized for RP Overhead, Workload Execution, and RP Idle across various scales of tasks/cores.](chart_image)
**RADICAL-EnTK: Building Blocks for Workflows**

- **Ensemble Toolkit (EnTK):** Toolkit to manage complexity of resource acquisition and application execution for scalable ensemble-based applications.

- **Design:**
  - User facing components (blue)
  - Workflow management components (purple) to manage the execution order of the individual tasks of the application
  - Workload management components (red) to manage resources and task execution via a runtime system (green)

- **Integrate with existing tools:**
  - Provides generic building block components that encourage a lego-style application creation
RADICAL-EnTK: Power to the Many

- **PST Programming Model:**
  - Task: an abstraction of a computational process and associated execution information
  - State: 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” has been executed
  - Design: Simplicity with performance
    - Simple programming model (P-S-T model)
    - Workflow Management Layer: (i) AppManager, (ii) WFProcessor
    - Workload Management Layer: ExecManager
    - Defined execution model and interfaces with different runtime systems

- **Support novel tools and libraries:**
  - EnkT used by many workflow systems (HTBAC, ExTASY, RepEx…)
RADICAL-EnTK: Performance (Titan)

The graph shows the execution time (seconds) and the number of failed tasks for different numbers of concurrent tasks and nodes. The x-axis represents the number of concurrent tasks divided by the number of nodes. The y-axis for execution time is in seconds, and the y-axis for the number of failed tasks is on a logarithmic scale.

Execution time (seconds):
- 2^0/384: 4000
- 2^1/768: 3000
- 2^2/1536: 2000
- 2^3/3072: 1000
- 2^4/6144: 0
- 2^5/12288: 0

Number of failed tasks:
- 2^9
- 2^10
- 2^11
- 2^12

The graph uses different colors to represent the number of tasks:
- 1 task: blue
- 2 tasks: orange
- 4 tasks: green
- 8 tasks: purple
- 16 tasks: yellow
- Failed tasks (right): gray
HTBAC: High-throughput Binding Affinity Calculator

- Python library for defining and executing ensemble-based biosimulation protocols
  - Protocols expressed and implemented using HTBAC’s API
  - HTBAC utilizes RADICAL-Cybertools (RCT): EnTK and RP

- Implemented and tested with **ESMACS** and **TIES protocols**

- Define additional adaptivity parameters that are passed down to the underlying runtime system.

- **TIES** (alchemical protocol) employs enhanced sampling at each lambda window to yield reproducible, accurate and precise relative binding affinities.

- **ESMACS** (endpoint protocol) is a computationally cheaper, but less rigorous method, it is used to directly compute the binding strength of a drug to the target protein from MD simulations (as opposed to differences in affinity).
Adaptive Quadratures in Binding Free Affinity

The uncertainty in the computed observable - measured using the standard error of the mean (SEM)

- Adaptive quadratures increase rate of convergence by reducing SEM faster than non-adaptive

Adaptive quadrature of the function $f(\lambda) = \partial U/\partial \lambda$ in the interval $[0, 1]$ using the trapezoidal rule.
- From left to right the simulations are increased to increase fidelity, with extra runs bisecting points where deviation between existing points is above a set threshold.
- The true integration error is the difference between the interpolated function and the actual function (shaded area).

- Adaptive quadrature algorithm adds additional simulations to reduce error on binding free affinity.
TIES Protocol

TIES (alchemical protocol) employs enhanced sampling at each lambda window to yield reproducible, accurate and precise relative binding affinities.
Observed accuracy of adaptive vs. non-adaptive in same simulation duration (6ns)

- Resource consumption decrease
- Error decrease
The graph shows the error in kcal/mol for different protein-ligand systems. The systems compared are:

- PTP1B L1-L2
- PTP1B L10-L12
- TYK2 L7-L8
- TYK2 L4-L9
- MCL1 L32-L38

The graph compares non-adaptive and adaptive methods. The non-adaptive errors are represented by blue bars, while the adaptive errors are represented by orange bars. The highest error is observed in the PTP1B L1-L2 system using the non-adaptive method.
SCALE Challenge Award 2018
Winner
Jumana Dakka, Kristof Farkas-Pall, Vivek Balasubramanian, Matteo Turilli, Shunzhou Wan, David W. Wright, Stefan Zasada, Peter V. Coveney and Shantenu Jha
with
"Enabling Trade-off Between Accuracy and Computational Cost: Adaptive Algorithms to Reduce Time to Clinical Insight"

Sandra Gesing
Chair of IEEE SCALE 2018
May 2018

Amy Apon
Chair of IEEE SCALE 2018
May 2018
Adaptive Ensemble Execution at Scale

- Adaptivity: TG not fully specified prior to execution; modification of TG based on runtime data generation.
- Execution Model for Adaptive TG:
  1) Encode application using known TG
  2) Traverse TG identify execution-ready
  3) Tasks executed
  4) Notification of a completed task (control-flow) or generation of intermediate data (data-flow) to evaluate and execute TG adaptations.
- Three types of adaptivity:
  - Task-count: number of tasks
  - Task-order: task dependency order
  - Task-attribute:
Adaptive Sampling: Expanded Ensemble

- Use for multiple distinct biomolecular adaptive workflows
- Expanded Ensemble:
  - MBAR estimate of the pooled data, and the std. deviation of the non-pooled MBAR estimates of four 200 ns fixed weight expanded ensemble simulations
  - Method 1: one single simulation
  - Method 2: multiple simulations with no analysis
  - Method 3: multiple simulations with local analysis
  - Method 4: multiple simulations with global analysis

Work with Kasson, Shirts
https://arxiv.org/abs/1804.04736
Summary

- Importance and diversity of “workflows” set to increase
  - Proliferation of middleware systems for “workflows” unsustainable
  - Substitute discussions of software with abstractions & execution models

- Building blocks approach to workflows
  - Focussed, principled design and development of middleware systems
  - Each building block has well defined performance characterization

- Algorithmic and methodological advances are needed
  - Adaptive execution of large ensembles
  - Multiple types of adaptivity at scale
Thank You!