Middleware Building Blocks for Workflow Systems

The goal of this paper is to contribute in the design of workflow systems that share capabilities but not limit functionality or performance, in order to help developers spend less time in such task, and focus more in their own development in a collective fashion. This is where the Building-Block approach comes in. Applying modularity at the system level and keeping the four design principles of self-sufficiency, interoperability, composability and extensibility in mind, RADICAL-CyberTools were developed.

These Tools are a set of software systems that can be used independently and integrated into middleware. Currently, there exists EnTK, Pilot, SAGA and WMS which is still under development. Each system defines its own entities, functionalities, states, events and errors.

Finally, the paper implements ways of integrating these tools into existing workflow systems like Pegasus, Swift or even Frameworks like Hadoop and Spark in order to overcome specific challenges on HPC machines.

Note: The focus is on the design approach, not the Tools!

RADICAL-Cybertools: Middleware Building Blocks for Scalable Science

With the present paper, the focus shifts now from the design approach perspective to a brief Technical introduction of the RTC, mainly the RP and EnTK, since RS is described in another work. Let's start with RP.

RP is a Python implementation of the pilot paradigm and architectural pattern. Basically two abstractions: Pilot and Compute Unit (task). With

These systems, users can submit pilot jobs to computing infrastructures and then use the resources acquired by the pilot to execute one or more tasks.

EnTK is also a Python implementation but of a workflow engine. It is in charge of supporting the programming and execution of applications with ensembles of tasks. Here, three abstractions are implemented: task, stage and pipeline

All in all, RTC are designed to work individually and as integrated system, with or without third party systems. RP decouples resource acquisition from task execution, while EnTK allows to codify ensemble applications as pipelines of stages of tasks.

Extensible and Scalable Adaptive Sampling on Supercomputers

Using software now as a tool for a domain-specific application, the present work is about reducing computational resources and simulation times for biological processes such as protein folding and protein dynamics. This is achieved through adaptive sampling, an iterative process where MD simulation from previous iterations are analyzed, and based on the analysis, a new iteration of shorter MD trajectories is initiated.

The ExTASY Framework is used, using Markov state Models with different restarting strategies. To be exact, two strategies combined seemed to yield better results: cmacro until the folded state is reached, continuing with cmicro to improve accuracy of the protein dynamics.

As a domain-specific workflow system, ExTASY makes use of RTC's EnTK and RP in order to acomplish a speed up of about one order of magnitud, compared to brute force MD simulation.

Concurrent and Adaptive Extreme Scale Binding Free Energy Calculations

The current work is about the use of adaptive approaches on ensemble-free energy protocols for drug discovery and design, with the goal of reducing computational resources and improving efficiency.

Among these protocols, we have ESMACS and TIES, both designed to use an ensemble MD simulation, with a focus on intra-protocol adaptivity which relies on intermediate runtime results within a protocol instance to define the following set of simulations, just like the previous paper. For this, the authors introduce the software HTBAC that relies on RTC's EnTK and RP, and is implemented in Python as a domain-specific library.

EnTK derives a resource request, converting it into a pilot description for RP. RP converts this pilot requests into a batch script that is submitted to the specified HPC. Once the pilot becomes active, EnTK identifies those application tasks that have satisfied dependencies and can be executed concurrently. EnTK's own Execution Manager uses RP to execute those tasks on pilot's resources.

Experiments executing ESMACS and/or TIES protocols were made, as well as with two adaptive methods: adaptive quadrature and adaptive termination. They show near-ideal weak and strong behavior for ESMACS and TIES, individually and together. Overall, accuracy, execution time and resources are improved by using the adaptive TIES protocols as well, positively contributing to the ensemble-based binding affinity ongoing research.