RADICAL Laboratory, Electric and Computer Engineering, Rutgers University, New Brunswick, NJ, USA

I. Introduction

Recent developments in both algorithms and architectures (in particular GPGPUs) have led to increasing interest in the use of molecular simulation to estimate the strength of macromolecular binding free energies [**?**]. However, for molecular simulations to truly influence decision making in industrial and clinical settings, the dual challenges of scale (concurrent screening of drug candidate using thousands of concurrent multi-stage pipelines) and sophistication (adaptive sampling or selection of binding affinity protocols based upon system behavior and statistical uncertainty) will need to be tackled.

Tools that facilitate the automated scalable and sophisticated computation of varied binding free energy calculations on high-performance computing resources are necessary. We have previously developed a tool for automating system building and simulation input generation, the Binding Affinity Calculator (BAC) [**?**]. More recently we have introduced the High-Throughput Binding Affinity Calculator (HTBAC) [**?**], which brings advances in the design of domain-specific workflow systems using building blocks to facilitate the automation of deployment and execution of rapid and accurate calculation of binding affinities on leadership class machines. In Dakka *et al.* [**?**] we demonstrated how HTBAC scales almost perfectly to hundreds of concurrent multi-stage pipelines for a single protocol.

In this paper we reproduce results from a collaboration project between UCL and GlaxoSmithKline to study a congeneric series of drug candidates binding to the BRD4 protein (inhibitors of which have shown promising preclinical efficacy in pathologies ranging from cancer to inflammation) [**?**]. These studies employed two different protocols, known as TIES and ESMACS [**?**], both based on multiple simulations of the same system.

TIES is rigorous but computationally expensive and has a limited range of applicability. ESMACS is approximate but can be employed across any set of ligands at lower computational cost. Drug design projects have limited resources, so initially large numbers of compounds must be cheaply screened to eliminate poor binders (using ESMACS), before more accurate methods (such as TIES) are needed as good binders are refined and improved.

In order to support such investigations, in addition to being scalable, HTBAC must be enhanced to support flexible resource reallocations schemes where resources can be moved between simulations run using different protocols or systems, for example, when one calculation has converged whilst another has not. This adaptability makes it easier to manage complex programs where efficient use of resources is required in order to achieve a time to completion of studies comparable to those of high throughput chemistry.

In this work we demonstrate the use of HTBAC to adaptively run both protocols, including mixed protocol runs. Calculations using either protocol require simulations of free and protein bound ligand necessitating the coordination of runs with heterogeneous computational requirements. Further to the use of a common framework to improve the ease of deployment and efficiency of execution of existing protocols we show how HTBAC can aid the development of enhanced approaches.

These developments fit into a wider vision in which the use of flexible and responsive computational protocols produce accurate, precise and reproducible estimates of the free energy of binding with meaningful error bars. Not only would this allow for wider uptake of computational techniques in industrial settings but opens up possibilities of using these technologies in clinical decision support scenarios. By creating a 'digital twin', where the target protein is based on the real patients genetic sequence, a specific individuals response to different treatments could be predicted.

The novel contributions of HTBAC are: (i) Unprecedented throughput: it allows the concurrent screening for drug binding affinities of multiple compounds at unprecedented scales, both in the number of candidates and resources utilized; (ii) Agile selection of different binding affinity protocols: HTBAC supports inter-protocol adaptivity, leading to resources being assigned at runtime to the "optimal" protocol (as determined by accuracy for given computational cost); (iii) Support for intra-protocol adaptivity, which provides the efficient execution of individual protocols.

In this paper we describe the background to our binding free energy protocols and how we designed HTBAC to meet the challenges faced in bringing this approach up to extreme scale. We then describe a set of experiments characterizing the performance and scalability of HTBAC and the development of an adaptive version of TIES.

II. Related Work

The strength of ligand binding is determined by a thermodynamic property known as the binding free energy (or binding affinity). One promising technology for estimating binding free energies and the influence of protein and ligand composition upon them is molecular dynamics (MD) [**?**]. A diversity of methodologies have been developed to calculate binding affinities MD sampling [**?**] and blind tests show that many have considerable predictive potential [**? ?**]. The development of commercial approaches that claim accuracy of below 1 kcal mol^{-1} has led to increased interest from the pharmaceutical industry [**?**]. The same technologies have also shown promise in analysing the influence of protein mutations on drug binding [**? ?**]. These developments have also spurred renewed

interest in the factors affecting the performance of different free energy methods [**? ? ?**]. As these developments combine with increased computational power it is becoming possible to integrate the information gained from large numbers of simulations to provide more predictive models using machine learning approaches [**?**]. It is this approach of combining predictive modeling with integrative technologies which provides one of the most promising avenues for developing decision support tools for drug discovery and clinical support.

As binding free energy approaches based on MD become more widely used, particularly by non-specialists, automated tools for system and simulation preparation have been developed [**? ? ?**], including our own tool the binding affinity calculator (BAC) [**?**]. Using BAC, we have designed two protocols with the demands of clinical decision support and drug design applications in mind: ESMACS (enhanced sampling of molecular dynamics with approximation of continuum solvent) [**?**] and TIES (thermodynamic integration with enhanced sampling) [**?**]. The former protocol is based on variants of the molecular mechanics Poisson-Boltzmann surface area (MMPBSA), which is an "approximate" endpoint method [**?**]. The latter on the rigorous "alchemical" thermodynamic integration (TI) approach [**?**].

Using these protocols, we have demonstrated the lack of reproducibility of individual simulations for a variety of protein systems, with calculations for the same protein-ligand combination using almost identical initial conditions producing widely varying results (binding affinities varying by up to 12 kcal mol $^{-1}$ for small molecules, whilst flexible ligands can vary even more) [**? ?**]. Indeed, our work has revealed how completely unreliable single simulation based approaches are. We have shown that averaging across multiple runs reliably produces results in agreement with previously published experimental findings [**? ? ? ? ? ?**], and correctly predicted the results of experimental studies performed by colleagues in collaboration [**?**]. We term this approach ensemble molecular dynamics, "ensemble" here referring to the set of individual (replica) simulations conducted for the same physical system.

Both TIES and ESMACS protocols are ensemble based. Basing these computations on the direct calculation of ensemble averages facilitates the determination of statistically meaningful results along with complete control of errors. In addition to accuracy and precision, in order for MD simulations to attain widespread use in industrial (and potentially clincal) settings, it is necessary that results can be obtained soon enough in order that the results can feed into decision making processes. These motivations provided the impetus behind our automated model building and simulation input generation package, the binding affinity calculator (BAC) [**?**]. As we scale up to larger datasets the use of the ensemble approaches will increasingly necessitate the use of middleware to provide reliable coordination and distribution mechanisms with low performance overheads. These considerations provide the motivation behind the development of HTBAC.

III. Science Drivers

The basic use case for HTBAC is to enable large scale free energy studies of protein-ligand binding using ensemble simulations. We have already briefly introduced two free energy calculation protocols, ESMACS and TIES [**? ?**], with reference to the comparative rigour of the former relative to the later. Here we provide more details of their practical specifications and of the protein system we used to benchmark and refine them in this work.

A. ESMACS and TIES

In both protocols, ensembles of MD simulations are employed to perform averaging and to obtain tight control of error bounds in our estimates. In addition, the ability to run replica simulations concurrently means that, as long as sufficient compute resources are available, turn around times can be significantly reduced compared to the generation of single long trajectories. Due to their shared philosophical underpinning both protocols share similar middleware requirements.

The current implementation of both TIES and ESMACS uses the the NAMD package [**?**] to conduct the simulations. Conceptually, each replica simulation consists of three stages: minimization, equilibration and production MD. In practice the equilibration phase is broken into multiple steps to ensure that the size of the simulation box does not alter too much over the simulation. During these steps positional constraints are gradually released from the structure and the system is heated to a physiologically realistic temperature. Whilst both protocols share a common workflow for individual replicas, the make up of the ensemble is different.

TIES is based on rigorous, but computationally expensive, calculations of relative free energies (i.e. results provide a comparison between two drugs). ESMACS, in contrast, provides absolute binding free energies at low computational cost, but to achieve this coarse grains many of the details of the system being studied. The simplifications employed by ESMACS can reduce its effectiveness in some systems.

In the case of ESMACS, an ensemble consists of a set of identical simulations differing only in the initial velocities assigned to each atom. Upon completion of the MD simulation, free energy computation (via MMPBSA and potentially normal mode analysis) is performed using AmberTools.

TIES is a so called "alchemical" method in which the fact that the potential used to decribe the system is user definable to transform one system into another. This allows for the calculation of free energy differences between the two systems, such as those induced by an alteration to a candidate drug. Typically a transformation parameter (called λ) is defined such that as it increases from zero to one the system description transforms from containing an initial drug to a target compound via a series of hybrid states. Sampling along λ is then required in order to compute the difference in binding free energy. Consequently, whereas in ESMACS all replicas sample using the same system description, in TIES sub-ensembles are executed at different points along λ. The change in free energy associated with the transformation is calculated by numerically integrating (via adaptive quadrature)

the values of the $\partial U/\partial\lambda$ across the full set of λ windows simulated. The TIES protocol, for example, relies on the numerical integration of $\partial U/\partial\lambda$.

Obtaining accurate and precise results from TIES requires that the λ windows correctly capture the changes of $\partial U/\partial\lambda$ over the transformation. This behavior is highly sensitive to the chemical details of the compounds being studied and varies considerably between systems. Typically, windows are evenly spaced between 0 and 1 with the spacing between them set before execution at a distance determined by the simulator to be sufficient for a wide range of systems. Typically, a TIES ensemble 65 replicas evenly distributed between 13 λ windows In order to obtain a meaningful TIES result it is necessary to not only simulate the drug pair in the protein but also in an aqueous environment, adding a further 65 replicas albeit it using a smaller system at lower computational cost.

Following simulation (and in the case of ESMACS free energy analysis steps) both protocols require the execution of short serial steps to provide summary statistics. Both protocols can are highly customizable, for example the number of simulation replicas in the ensemble and the lengths of their runs can be varied. More generally though, there may be cases where it is important to increase the sampling of phase space possibly through expanding the ensemble or in TIES changing the distribution of λ windows. In addition, the ESMACS protocol can also be extended to account for adaptation energies involved in altering the conformation of the protein or ligand during binding through the use of separate component simulations.

B. The Value of Adaptivity

Both ESMACS and TIES have been successfully used to predict binding affinities quickly and accurately. Nonetheless, they are very expensive computationally, and optimizing the execution time while still improving the accuracy is desirable. Given the very large number of drug candidates, it is imperative to gain maximum insight into potential candidate compounds using time and resources efficiently. This provides one clear motivation for the use of adaptive methods which minimize the compute time used whilst producing binding free energy estimates meeting pre-defined quality criteria (such as convergence or statistical uncertainty below a given threshold).

A second driver for adaptivity is that such algorithmic methods will typically involve compounds with a wide range of chemical properties which can impact not only the time to convergence, but the type of sampling required to gain accurate results. In general, there is no way to know before running calculations exactly which setup of calculation is required for a particular system. For example in TIES, the number (or the exact location) of the λ windows that will most impact the calculation are not known *a priori*, and change between physical systems (drugs). As multiple simulations must be run for each window, sampling with a very high frequency is expensive and impractical. Furthermore, adaptive placement of λ windows is likely to better capture the shape of the $\partial U/\partial\lambda$ curve, leading to more accurate and precise results for a given computational cost.

On occasion, alchemical methods may be very slow to converge; in such circumstances use of another method, such as ESMACS, may be the best option. This means that the most effective way to gain accurate and precise free energy results on industrially or clinically relevant timescales is to be able to adapt both sampling (intra-protocol) and even the type of calculation (inter-protocol) used at run time. With potentially thousands of simulations, often employing multiple analysis methodologies, this provides the most effective way to utilize these techniques and resources at scale.

C. Target protein: BRD4

Bromodomain-containing proteins, and in particular the four members of the BET (bromodomain and extra terminal domain) family, are currently a major focus of research in the pharmaceutical industry. Small molecule inhibitors of these proteins have shown promising preclinical efficacy in pathologies ranging from cancer to inflammation. Indeed, several compounds are progressing through early stage clinical trials and are showing exciting early results [**?**]. One of the most extensively studied targets in this family is the first bromodomain of bromodomain-containing protein 4 (BRD4-BD1) for which extensive crystallographic and ligand binding data are available [**?**].

We have previously investigated a congeneric series of ligands binding to BRD4-BD1 (we shall from now on refer to this are simply BRD4) using both ESMACS and TIES. This was performed in the context of a blind test of the protocols in collaboration with GlaxoSmithKline [**?**]. The goal was to benchmark the ESMACS and TIES protocols in a realistic drug discovery scenario. In the original study, we investigated chemical structures of 16 ligands based on a single tetrahydroquinoline (THQ) scaffold. Here we focus on the first seven of these ligands to test and refine the protocols used and the way in which they were executed. The results of our previous work provide a benchmark of both accuracy and statistical uncertainty to which we can compare our new results.

Initial coordinates for the protein-ligand system were taken from the X-ray crystal structure PDB ID: 4BJX. This structure contains a ligand based on the THQ template and other ligands were alligned with this common scaffold. Preparation and setup of the simulations were implemented using our automated tool, BAC [**?**]. This process including parametrization of the compounds, solvation of the complexes, electrostatic neutralization of the systems by adding counterions and generation of configurations files for the simulations. The AMBER ff99SB-ILDN force field was used for the protein, and TIP3P was used for water molecules. Compound parameters were produced using the general AMBER force field (GAFF) [**?**] with Gaussian 03 to optimize compound geometries and to determine electrostatic potentials at the Hartree–Fock level with 6-31G** basis functions. The restrained electrostatic potential (RESP) module in the AMBER package was used

to calculate the partial atomic charges for the compounds. All systems were solvated in orthorhombic water boxes with a minimum extension from the protein of 14 Å (the resulting systems contain approximately 40 thousand atoms).

IV. HTBAC: Design and Implementation

Most advances in high-performance computing have focused on the scale, performance and optimization of single long simulations. Due to the end of Dennard scaling and methodological advances, many applications are now formulated using multiple, shorter ensemble-based simulations. There are however, limited software solutions that can support the scalable execution of multiple tasks concurrently.

HTBAC has three primary design requirements: (1) enable the scalable execution of concurrent multi-stage pipelines, flexibly partitioned across different protocols, (2) abstract the complexity of building protocols, execution and resource management; and, (3) provide adaptive features (i.e., modifying the task-graph during runtime, based on previous tasks), to enable efficient protocols and effective resource sharing across protocols, without explicit resource management requirements.

HTBAC builds upon established middleware solutions [**?**], validated runtime abstractions [**?**] for scalable execution of heterogeneous computational tasks, and customizes them for the computation of binding free energies. HTBAC derives many of the advantages of a lightweight, flexible domain specific workflow layer from its use of RADICAL-Cybertools (RCT) which are functionally well-defined and delineated middleware building blocks. RCT [**?**] are engineered to support extensible and scalable workflows across diverse computing platforms. The two primary RCT components that HTBAC depends upon are the Ensemble Toolkit (EnTK) and RADICAL-Pilot (RP). Figure 1 shows the layered architecture of HTBAC, EnTK, and RP.

EnTK simplifies the process of creating ensemble-based applications with complex coordination and communication requirements. The EnTK API exposes the PST model which consists of three components: **Pipeline**, **Stage**, and **Task**. HTBAC promotes binding affinity **Protocols** as the user-facing construct, and also uses the programming model defined by the Pipeline, Stage and Tasks model to express a variety of protocols: each stage is a computational **task**, and the ordered aggregation of these stages alongside their dependencies as a **pipeline**. A workflow is comprised of N_P instances of the P^{th} protocol.

The **Protocol** class can be instantiated as one of two protocols: ESMACS or TIES. For TIES, each protocol typically corresponds to a single physical system, and multiple instances can be supported. The TIES protocol object enables the user to select a physical system, number of replicas, core allocation per replica, and λ windows to sample. Figure 2 provides the visual implementation of the TIES protocols into the PST model.

Once the workflow is described, it is submitted to EnTK's **Application Manager** which sets up multiple processes, threads and a RabbitMQ message queue for communication.

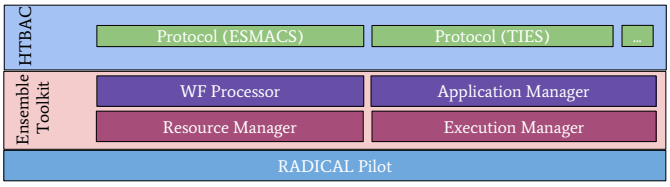

Fig. 1. Layered architecture of HTBAC, EnTK, and RP. The HTBAC API exposes the Protocol component. Current protocols supporting the use cases include ESMACS/TIES. EnTK serves as the workflow execution system, and by managing the workflow and workload. RADICAL-Pilot serves as the runtime system.

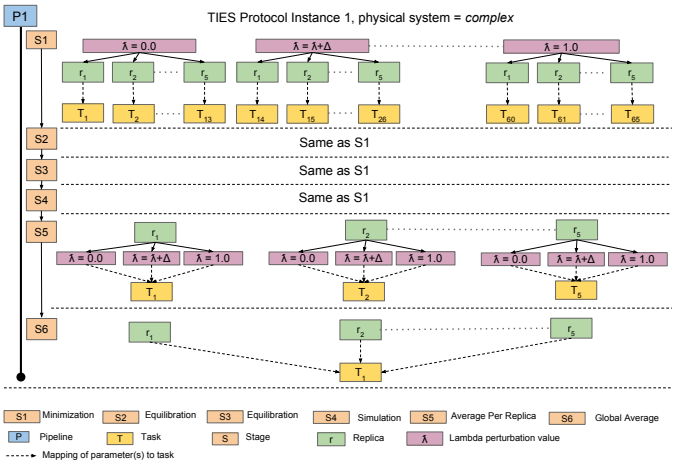

Fig. 2. TIES protocol expressed using the EnTK PST model. Each protocol instance maps to a single Pipeline, comprised of Stage(s) which maintain temporal order. Each Stage executes n tasks, where n represents the number of unique lambda-replica combinations.

EnTK identifies tasks which have satisfied dependencies and can be executed concurrently. EnTK's **Execution Manager** uses the underlying runtime system, RADICAL-Pilot to execute the tasks on specific target resources.

In Section II we highlighted how an ensemble simulation approach can both aid sampling and improve uncertainty quantification for free energy calculations. Despite these crucial advantages, it remains non-trivial for field researchers to write biosimulation applications that involve individual protocols supporting multiple replicas, and by extension multiple protocols. With HTBAC, this burden is minimized by specifying the number of concurrent instances of the **Protocol** object. Moreover, the ability to generate multiple protocol instances enables the user to investigate a range of physical systems (i.e., drug candidates) concurrently.

HTBAC allows the simple expression and concurrent execution of multiple distinct protocols, thereby enabling concurrent screening of drug candidates. HTBAC not only simplifies the expression of complex binding affinity protocols, but also provides hitherto unavailable capabilities, viz., the adaptive execution of these protocols, without any additional programming burden.

In section III, we described a particular type of adaptivity within TIES that would enable the simulation to reach con-

vergence earlier. Adaptive execution requires changes to the taskgraph during runtime, capabilties that are supported by the RCT runtime layers [**?**].

The TIES adaptive workflow decides, post-mortem, about the placement of new lambda windows for the next set of simulations. In turn this requires task- count adaptivity which we define as runtime changes to the number of tasks in the taskgraph. HTBAC monitors the output results from the completed tasks during runtime, and redefines the existing workflow by adding more λ windows. Using EnTK, HTBAC creates the necessary set of new tasks. The **Application Manager** in EnTK signals RP to bypass termination of the pilot and instead keep the session alive, as long as additional tasks are submitted. As the number of tasks grow during runtime, the ratio of core-to-task will fluctuate. HTBAC exposes the core allocation per task, and allows the user control to pre-specify resource requirements of additional tasks.

V. Experiments

1) Experiment Setup: We perform weak and strong scalability experiments on NCSA Blue Waters–a 13.3. petaFLOPS Cray, with 32 Interlago cores/50 GB RAM per node, Cray Gemini, Lustre shared file system. Our experiments are launched on Blue Waters and monitored using a persistent virtual machine, as Blue Waters does not allow for executing applications directly on the login node.

We used HTBAC version 0.1 EnTK version 0.6 and RP version 0.47. The MD engine used is NAMD-MPI for tasks pertaining to $S1$ - $S4$, while the analysis stage, $S5$ use AmberTools. For both adaptive and nonadaptive experiments, the minimization tasks of $S1$ are assigned 1000 steps, while the equilibration tasks in $S2$ and $S3$ are assigned 5000 steps. In the nonadaptive experiments, the production simulation tasks in $S4$ are assigned 50,000 steps. For the adaptive experiments, each substage of $S4$ i.e. $S4.1$ - $S4.4$ is assigned 50,000 steps.

For weak scaling performance, we use between 4160 and 33280 cores as indicated in Figure 3 because the NAMD executable used in all tasks from $S1$-$S4$ require at least 32 cores per task. From our own scalability performance measurements of NAMD on Blue Waters, we observe the ideal cores per task to be 16, however Blue Waters does not permit running multiple MPI applications on the same node, hence each NAMD task requires a full node to maintain concurrency.

For strong scaling performance, we fix the number of protocol instances to 8 instances while varying the amount of resources as demonstrated in Figure **??**.

For the TIES protocol, each pipeline consists of six stages. Each of the simulation stages contains a task for every unique (λ, replica) combination. In the non-adaptive workflow scenario, the first 11 λ windows consist of the following values: L is a vector with

$$L = \{x_i : x_i \in [0,1] \; and \; x_{i+1} = x_i + \delta\}, where \; \delta \; is \; 0.1. \tag{1}$$

We append two additional values on both ends of L, completing 13 λ windows. Each λ window consists of five

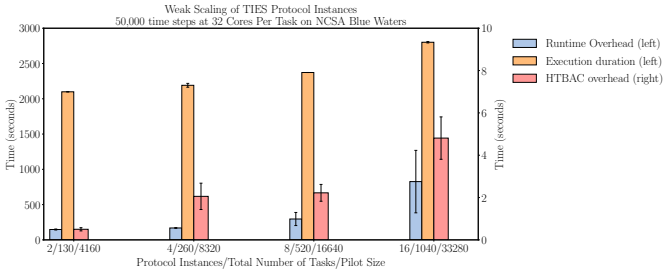

Fig. 3. Weak scaling properties of HTBAC. We investigate the weak scaling of HTBAC as the ratio of the number of protocol instances to resources is kept constant. Overheads of HTBAC (right), and runtime overhead (left) and TTX (left) for experimental configurations investigating the weak scaling of TIES. We ran two trials for each protocol instance configuration. Error bars in TTX in 2 and 8-protocol runs are insignificant.

replicas. Therefore there are a total of 65 tasks for every simulation stage. The production simulations stage, $S4$ as described in figure 2 executes a 4 ns simulation duration. The analysis stages of the protocol reduce the number of tasks. The first analysis task consists of five tasks where each task performs an aggregate analysis over all λ windows for each replica. The second analysis stage consists of one task that aggregates the previous results and computes a single average across all replicas.

2) Scaling and Performance Characterization: We first characterize the overheads of HTBAC and the runtime system. HTBAC enables concurrent execution of multiple protocol instances. With each new protocol instance generated for an application, the HTBAC overhead grows to match the additional requirement of generating new protocols. The total time to completion (TTC) can be expressed as following: $TTC = TTX + T_O$ where TTX measures the execution duration across all task, including file staging, MD kernel execution, and post-executables, and T_O the total overhead is given by the sum of the constituent overheads:

$$T_O = {T_O}^{\text{HTBAC}} + {T_O}^{\text{EnTK}} + {T_O}^{\text{RP}}$$

We investigated the weak scalability properties for the TIES protocol by growing the number of protocol instances while adhering to the required number of pipelines. By design of each protocol, an increase in the number of instances simply means an increase in the number of pipelines. The first weak scalability experiment demonstrates the behavior of HTBAC, EnTK and RP using the multiple instances of the TIES protocol. By design of weak scaling, the ratio between the number of pipelines and cores are kept constant. As the number of cores (measure of resource) changes by a factor of 2, we introduce twice as many protocol instances. As designed, the weak scaling property provides insight into the size of the workload that can be investigated in a given amount of time.

The HTBAC overhead shows a super linear increase as we grow the number of protocol instances. However, the HTBAC overhead is a minimal contributor to TTX. The runtime overheads, mainly EnTK and RP, depend on the number of tasks that need to be translated in-memory from a Python object

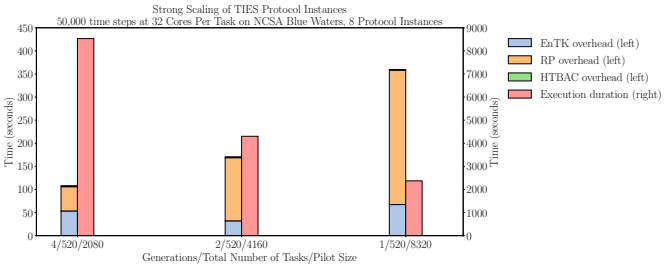

Fig. 4. Strong scaling properties of HTBAC. We investigate the strong scaling of HTBAC with a fixed number of protocol instances while varying the amount resources. Overheads of HTBAC and runtime overhead (left) and TTX (right) for experimental configurations investigating the strong scaling of TIES.

to a compute unit (CU) description. As such, it is expected to grow proportionally to the number of tasks. EnTK submits CU descriptions to a MongoDB used by RP. RP pilot pulls these descriptions from the same database. In addition, each stage constructed by EnTK maintains sequential barriers. RP remains dormant until completion of the current tasks before staging the next tasks. The impact of the synchronization barriers increases with the number of CUs as seen in the 16 protocol instances data point in 3.

Furthermore, an additional overhead, driven by the aprun launch method, increases as we approach a system limit on the number of permissible aprun processes when scaling from 8 to 16 protocols, which translates to 520 and 1040 concurrent tasks, respectively. We account for the aprun failures in the TTX. Together, the EnTK-enabled synchronization barriers and aprun overhead failures introduce delays in the scheduling of the CUs and results in higher overheads. Lastly, we notice that each additional protocol instance contributes to roughly 55 additional seconds in TTX.

3) Strong Scaling Experiments: Next we repeat the same design of the weak scalability experiments but examine performance of strong scaling when fixing the number of pipelines and varying the resources. The comparison between weak and strong scalability demonstrates the overhead introduced by load balancing and scheduling tasks in multiple generations. As we scale the number of generation of concurrent tasks executions, we half the resources allocated by the pilot.

Furthermore, as we scale the number of generations, the runtime overhead of RP scales linearly. RP requires a proportional amount of time to schedule tasks to the scale of execution. The main contributor to the increase in overhead is derived by the time of resources inactivity while RP schedules new tasks. As such, the overhead is expected to grow proportionally to the number of concurrent tasks.

4) Validation Experiment: HTBAC fully automates the process of calculating the binding affinity of protein-ligand complexes from reading the input all the way to analysing the final results. In order to validate the correctness of the results, we have devised a set of experiments. These experiments are vital to gain confidence in the algorithm and to prove that it is indeed calculating the correct values.

The validation experiments were based on the original study of Wan et. al. [**?**]. We selected a subset of the protein ligand systems that were the subject of that study: they are the ligand transformations 3 to 1, 4, and 7. We then performed a full simulation on all 3 systems and calculated the binding affinity (see Table I) using HTBAC.

System	HTBAC	Wan et. al	Experiment
BRD4 **3 to 1**	0.39(10)	0.41(4)	0.30(9)
BRD4 **3 to 4**	0.02(12)	0.01(6)	0.00(13)
BRD4 **3 to 7**	−0.88(17)	−0.90(8)	−1.30(11)

TABLE I
COMPARISON OF THE CALCULATED BINDING FREE ENERGIES USING HTBAC, FROM THE ORIGINAL STUDY BY WAN ET. AL AND EXPERIMENTAL DATA. THE TWO THEORETICAL STUDIES USED THE SAME PROTCOL IN PRINCIPLE. THIS EXPERIMENT PROVED THAT HTBAC HAS INDEED IMPLEMENTED TIES CORRECTLY, AS THE CALCULATED VALUES ARE EITHER THE SAME OR WITHIN ERROR BAR OF THE ORIGINAL STUDY. ALL VALUES ARE IN **KCAL MOL^{-1}**.

The results show that all three $\Delta\Delta$G values are within error bars of the original study, reinforcing the fact that HTBAC has indeed correctly implemented the complex workflow of TIES.

5) Adaptive Experiments: The TIES workflow can benefit from an adaptive execution environment to improve the efficiency and accuracy of result. In *adaptive experiments* we implemented the adaptive quadrature algorithm specifically customized for biosimulations.

In the adaptive workflow, over the course of a protocol instance we alter the number of λ windows being simulated. The position of new λ windows depends on the estimated error of the integral measured between adjacent windows. Increasing the number of λ windows in regions of rapid change will increase the accuracy of the overall integral to a greater degree than an arbitrarily placed window. In order to adaptively add lambda windows, we need access to the $\partial U/\partial\lambda$ values during runtime. Therefore, we break down the single production simulation stage (S4) from the nonadaptive workflow into multiple smaller stages, each running for 1 ns. Once each simulation is complete within a stage a decision is made about whether more λ windows are required, and if so where they should be placed.

We start out the simulation with 5 replicas of *3* equally spaced λ windows, and equilibrate them. Then we repeatedly execute shorter production simulations followed by an analysis phase which determines where to place new lambda windows. This procedure is repeated until convergence, at which point all concurrent simulation are terminated. We define convergence as the point in the production-analysis loop at which a desired error threshold is reached.

The success of this algorithm is determined by the decision where additional λ windows should be introduced. In adaptive quadrature, this decision is made by calculating an error estimate on the integral and comparing this to a threshold value. Due to the stochastic nature of biosimulations, it is non-trivial to determine this error, and as a proof of concept we simplified this decision to replicate pre-calculated results. In future studies we plan to replace this with a dynamic decision process.

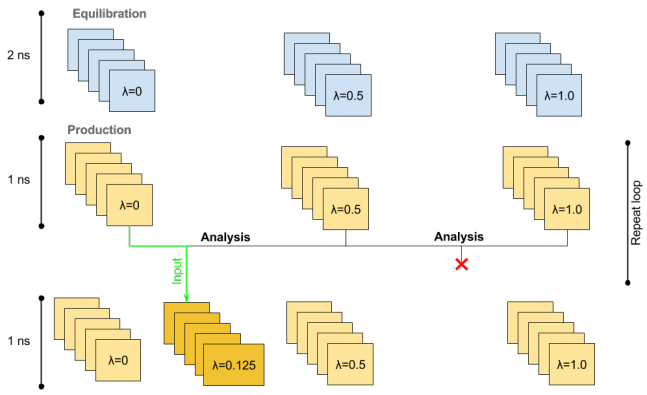

Fig. 5. Illustrating the adaptive workflow. After the 3 initial lambda windows are equilibrated, the first production stage starts. This is followed by analysis at every lambda interval, to decide whether to add a new window in the middle. The production-analysis is repeated for 4 production steps in our implementation, not shown here.

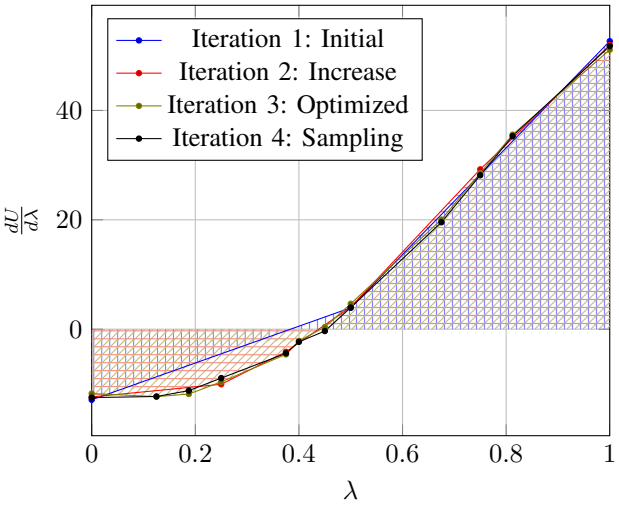

Fig. 6. Approximating the intergral under the curve, hence calculating ΔG. The adaptive algorithm reevaluates the efficiency of the lambda window mesh after every 1 ns and makes a decision whether to place more lambda windows inside certain ranges. As we iterate every 1 ns, the integral approximation becomes more accurate.

Inter-node communication introduces a constraint on the number of new lambda windows that can be added at each iteration. To reduce the overhead of inter-node communication, simulations must run on an integer number of nodes. This means that the number of new λ windows (i.e. the number of simulations) *has* to be either doubled or left unchanged. If the number of window is doubled the nodes per simulation can be halved automatically. Our prototype algorithm loops through the current λ window pairs until this criterion is reached, forcefully adding more if needed.

We introduce only a single degree of freedom relative to baseline "non- adaptive" experiments, thus our experiments implement adaptive change in the λ windows sampled and not the timing of execution. HTBAC provides the functional capability to adaptively determine the time at which the λ windows are chosen. However in this paper, we do not investigate the impact of such adaptivity, as the objective is to determine the feasibility of adaptive execution and resulting scientific merit of adaptive decision making.

6) Adaptive Quadrature Experiments Results: One way to visualize the adaptivity in this experiment is to plot the function and integral approximations at every iteration. Figure 5 shows how the value of ΔG (i.e. the integral of the function) for test system **3 to 7** improves as the algorithm places more lambda windows. It is imporant to note, that the new points are decided *a priori* for reasons discussed above but this is opaque to the algorithm. This is a proof that we have the adaptive capabilities and which pave the way for more advanced biosimulation algorithms.

VI. Discussion and Conclusion

Simulation protocols based on ensembles of multiple runs of the same system provide an efficient method for producing robust free energy estimate, and equally important statistical uncertainties. Variations in the chemical and biophysical properties of different systems impact the optimal protocol choice for different proteins and classes of drugs targetting them. However, the optimal protocol for a given system is difficult to determine *a priori*, thus requiring runtime adaptation to workflow executions. We introduce the High-throughput Binding Affinity Calculator (HTBAC) to enable the scalable, adaptive and automated calculation of the binding free energy on high-performance computing resources.

In this paper we demonstrate: (i) How HTBAC allows the concurrent screening for drug binding affinities of multiple compounds at unprecedented scales, both in the number of candidates and resources utilized. Specifically, we investigated weak scaling behaviour for screening sixteen drug candidates concurrently using thousands of multi-stage pipelines on more than 32,000 cores. This permits a rapid time-to-solution that is essentially invariant with respect to the calculation protocol, size of target system and number of ensemble simulations. (ii) The validation of binding free energies computed using HTBAC with both experimental and previously published computational results; (iii) HTBAC enabled the adaptive execution of the TIES protocol providing greater convergence (i.e., lower errors) for a given amount of computational resources. To the best of our knowledge, adaptive TI protocols have not been benchmarked against non-adaptive implementations.

As such, HTBAC advances the state of the scale and sophistication of binding affinity calculation. In addition to reporting increasingly sophisticated adaptive scenarios, in future, we will extend HTBAC to support the "design of experiments", facilitating optimization at the level of the overall computational campaign and time-to-insight for a large database of drug candidates, as opposed to for single simple calculations.