# Large Scale Binding Affinity Calculations on Commodity Compute Clouds

Zasada, S.J.[1], Coveney, P.V.[1,2,3]
[1]EnsembleMD Ltd, 10 Park Avenue, Woodford Green, Essex, IG8 0EU, UK.
[2]Centre for Computational Science, University College London, London WC1H 0AJ, United Kingdom.
[3]Computational Science Laboratory, Institute for Informatics, Faculty of Science, University of Amsterdam, Amsterdam 1098XH, The Netherlands.

## 1. Calculating Drug Binding Affinities

In recent years it has become possible to calculate the binding affinities of compounds bound to proteins via rapid, accurate, precise and reproducible free energy calculations. The ability to do so has applications from drug discovery to personalised medicine. This approach is based on molecular dynamics simulations and draws on sequence and structural information of the protein and compound concerned. Free energies are determined by ensemble averages of many molecular dynamics replicas, each of which requires hundreds of CPU or GPU cores, typically requiring a supercomputer class resources to be calculated on a tractable timescale. In order to perform such calculations there are also requirements for initial model building and subsequent data analysis stages.

In the past few years we have shown how to compute free energies via such ensemble averaging. Our approach uses two protocols known as ESMACS [1] and TIES [2]. ESMACS is centred on the molecular mechanics Poisson-Boltzmann surface area (MMPBSA) method, while TIES is built around thermodynamic integration. The protocols are complementary in that ESMACS is an absolute free energy method able to estimate binding affinity able to estimate binding affinities for highly diverse ligands of varying charges, whereas TIES is particularly suitable for estimating relative free energies of pairs of similar (congeneric) compounds and/or mutated protein sequences.

To automate the building and execution of the ESMACS and TIES protocols, we have developed a workflow known as the Binding Affinity Calculator (BAC) [3]. In this paper, we describe the approach we have taken to deploy and execute simulations on commercial cloud computing resources in a fully automated fashion, to reliably predict binding affinities on timescales relevant to the domains of application.

## 2. Gaining Access to Resources

On a present day high performance or cloud computing platform, with many thousands of cores, in the time it takes to run one calculation, we can do all fifty, so there is no need for extended wall clock time; hence the ability to predict on a clinical timescale is now entirely possible, given sufficient resources. The problem is the availability of such resources. Typically, access to the largest scales of high performance computing is available primarily to academic researchers via a competitive grant proposal process, or to government funded institutions. Furthermore, in normal settings of this kind jobs are scheduled through a batch queuing process, meaning that no guarantee can be given as to when any individual job will be run.

The nature of binding affinity calculation workflows means that the computations can be decomposed into ensembles of smaller molecular dynamics simulations. While each simulation requires typically 32 to 64 tightly interconnected compute cores, the overall ensemble of simulations behaves like an 'embarrassingly parallel' job, with no requirement for inter-process communication. As such, these types of ensembles map very closely to the architectures of commercially available compute clouds, which make available single compute nodes of up to 128 CPU cores and nodes with multiple GPUs. The on-demand nature of commodity cloud means that such resources can be called up when required and only the resources used are paid for.

## 3. Simulation Management

To enable the BAC workflow to be easily executed, we have developed ufBAC, which enables BAC to be run via a Software as a Service model, hiding from the user the complexities of the command line tools used to build models, executing them on computational resources, and analysing the results. ufBAC is intended to plug in to a range of computational back ends, from HPC provided by national research facilities, to commercial cloud platforms.

Selecting a cloud computing platform involves a trade-off between flexibility and vendor lock-in. While the basic commercial offering of the major cloud vendors is broadly similar, each vender offers their own value-added services on top of the basic platform. The more of these services that are used, the closer you are tied to a particular cloud vendor and the harder it is to switch platforms. We have taken an approach to limit the possibility of vendor lock-in by building on a common set of components that allow us to easily port our applications between clouds, namely Docker and Kubernetes.

Docker is a container system that allows us to encapsulate the legacy application components at the heart of the BAC workflow, making them portable and thus allowing them to be run on any cloud platform. Kubernetes is a related component that allows us to quickly spin up virtual clusters to run the dockerized applications. The architecture of the ufBAC cloud deployment is shown in figure 1.

Each execution of the BAC workflow on a particular cloud platform requires a data storage container to be created, to contain the input and output data generated by the different stages of the workflow. This is the only part of the ufBAC workflow that makes use of a platform specific feature. The initial model files must be uploaded to this storage container, before the first application component is run.

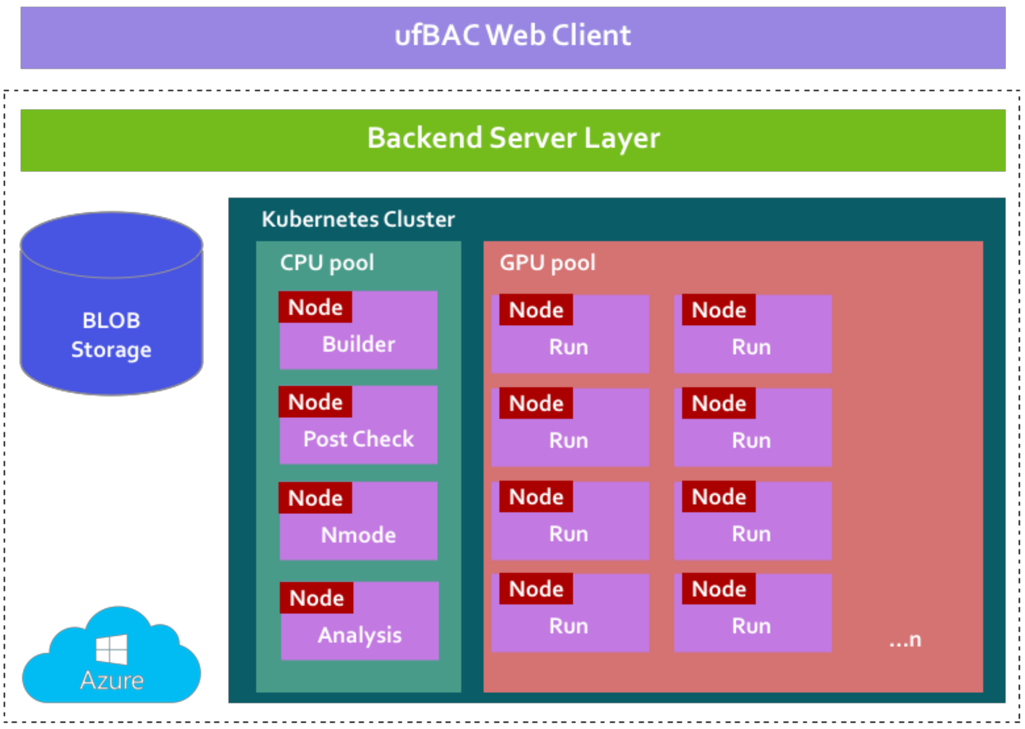

***Figure 1*** *The architecture of a ufBAC cloud deployment. The system is built around Docker and Kubernetes, and makes minimal use of the features available on a specific cloud platform.*

Each Docker container wrapping an application contains all of the libraries and supporting tools required to run the application. The Docker containers are stored in a Docker registry, from which they can be downloaded and run.

In order to run Docker containers, a Kubernetes cluster must be provisioned on the target cloud platform. This cluster is somewhat like a loosely coupled HPC cluster, comprising nodes of a specific type. Kubernetes is able to download and run *n* instances of a Docker container, relieving the user of the need to manually provision cloud resources.

Once the cluster is provisions, the ufBAC orchestration server uses the Kubernetes API interface to deploy and execute the containerized components of the workflow on the cluster. Results of the simulations are stored in the storage container after execution, from where they can be used to initiate the next step of the workflow or be analysed.

In this talk we will describe in detail the architecture and operating of the ufBAC system on commercial cloud platforms. We will also report on our experience of using our Azure deployment of the ufBAC system to perform large scale binding affinity calculations.

## 4. References

[1] Wan S, Knapp B, Wright DW, Deane CM, Coveney PV. Rapid, precise, and reproducible prediction of peptide-MHC binding affinities from molecular dynamics that correlate well with experiment. *J Chem Theory Comput* 2015, 11:3346-3356.
[2] Bhati AP, Wan S, Wright DW, Coveney PV. Rapid, Accurate, Precise, and Reliable Relative Free Energy Prediction Using Ensemble Based Thermodynamic Integration. *J Chem Theory Comput* 2017, 13:210-222.
[3] Sadiq, S. K.; Zasada, S. J.; Coveney, P. V. In *Computational Life Sciences II*; Berthold, M., Glen, R., Fischer, I., Eds.; Lecture Notes in Computer Science; Springer Berlin / Heidelberg, 2006; Vol. 4216; pp 150–161