Domain-Specific Energy Modeling for Drug Discovery and Magnetohydrodynamics Applications

Lorenzo Carpentieri
Marco D'Antonio
Kaijie Fan
Luigi Crisci
Biagio Cosenza
University of Salerno
Italy

Federico Ficarelli
Daniele Cesarini
CINECA
Italy

Gianmarco Accordi
Davide Gadioli
Gianluca Palermo
Politecnico di Milano
Italy

Peter Thoman
Philip Salzmann
University of Innsbruck
Austria

Philipp Gschwandtner*
Markus Wippler
PH3 GmbH
Austria

Filippo Marchetti
Daniele Gregori
E4 Computer Engineering
Italy

Andrea R. Beccari
Dompé Farmaceutici Spa
Italy

ABSTRACT

Over the past few years, the adoption of energy efficiency techniques in modern computer systems is becoming increasingly relevant for sustainable computing. A well-known power management software technique for energy-efficient computing is frequency scaling which modulates the device frequency to explore the energy-performance trade-off. To achieve energy savings, a frequency tuning phase is required because different applications can have different energy and runtime behaviors depending on the frequency setting. Machine learning models can be used to predict energy and runtime, and therefore optimal frequency configurations, based on static or dynamic features extracted from the target application. While general-purpose energy models can be very accurate for a wide range of applications, their accuracy can be limited by the specific input of the target application. We present an energy characterization that spans the fields of drug discovery and magnetohydrodynamics by using two real-world applications as case studies: LiGen and Cronos. Additionally, to overcome the limitations of general-purpose approaches, we define two domain-specific energy models, which enhance the general-purpose energy models by leveraging the target application's input parameter to increase the final accuracy. Experimental results show that for both applications, domain-specific models achieve a ten times lower error compared to the general-purpose energy models.

*Also with the University of Innsbruck.

CCS CONCEPTS

• Hardware → Impact on the environment; **Power estimation and optimization**; **• Computer systems organization → Heterogeneous (hybrid) systems**; **• Applied computing →** Computational biology; Physics.

KEYWORDS

Frequency scaling, Heterogeneous computing, Energy efficiency, Modeling

ACM Reference Format:
Lorenzo Carpentieri, Marco D'Antonio, Kaijie Fan, Luigi Crisci, Biagio Cosenza, Federico Ficarelli, Daniele Cesarini, Gianmarco Accordi, Davide Gadioli, Gianluca Palermo, Peter Thoman, Philip Salzmann, Philipp Gschwandtner, Markus Wippler, Filippo Marchetti, Daniele Gregori, and Andrea R. Beccari. 2023. Domain-Specific Energy Modeling for Drug Discovery and Magnetohydrodynamics Applications. In *Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis (SC-W 2023), November 12–17, 2023, Denver, CO, USA*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3624062.3624261

1 INTRODUCTION

The 2030 Agenda for Sustainable Development [33] sets important goals to improve the lives and prospects of all people, ranging from critical health issues to sustainable industrialization and energy efficiency. From a computational perspective, we are interested in computational approaches that have a small energy footprint, by providing full-stack measures to reduce the energy consumption of modern computing systems; but we also look for those big computational challenges that impact people's lives.

Energy-efficient computing has been driving research in many sectors, from engineering cooling systems to designing low-power hardware architectures. Hardware characteristics can be exploited by different techniques for reducing energy consumption through the use of software interfaces: power-capping limits the hardware's power over a time frame in order to achieve an overall power budget; near-threshold voltage computing is focused on working at lower voltages to increase energy efficiency [26]; Dynamic Voltage and Frequency Scaling (DVFS) [25] is a technique that aims to reduce power consumption by dynamically adjusting voltage and frequency. In this work, we use DVFS to reduce energy consumption in heterogeneous systems. Recent GPUs from AMD, Intel, and NVIDIA provide APIs for monitoring and managing the power state and changing the core frequency, i.e., ROCm SMI [2], Level Zero [24], and NVML [34], respectively. The availability of such energy capabilities paves the way for optimizations that focus not only on performance but also on the energy consumption of a GPU program. However, frequency scaling optimization requires accurate energy models: we can only save energy if we know which frequency configuration will give us a gain in energy consumption with minimal or possibly no performance loss.

State-of-the-art energy models [6, 15, 20, 21, 30, 43] propose the use of machine learning to predict the energy consumption of a given program. However, these models are rather general-purpose and may be inaccurate for some specific applications, in particular for problems where the model inaccuracy is due to varying input sizes. The key idea of this paper is that, if we target a specific application, we can build a much more accurate energy model. Therefore, we propose to move from a generic to a domain-specific energy model for higher accuracy.

To validate our modeling approach, we consider two applications that have a large impact on sustainable development goals. The first application is LiGen [3], a drug discovery platform by Dompé Spa that uses the GPU to accelerate the docking and scoring algorithm. LiGen is part of the EXSCALATE drug discovery platform, which recently identified a generic osteoporosis drug as an effective treatment for coronavirus [14]. The second application is Cronos [28, 29], a hydrodynamics and magnetohydrodynamics code developed primarily to solve various problems studied in the field of astrophysical modeling, but with a solver that also supports other conservation laws that can be provided by the user. Overall, this paper makes the following contributions:

- An energy characterization of drug discovery and magnetohydrodynamics real-world applications on AMD MI100 and NVIDIA V100;
- A methodology for building domain-specific energy models capable of predicting both energy and runtime;
- A comparison of the domain-specific models against a general-purpose state-of-the-art energy model on the two real-world applications, i.e., LiGen and Cronos.

The rest of this paper is organized as follows. Section 2 describes the motivations. Section 3 presents the energy characterization. Section 4 describes the domain-specific energy modeling methodology. Section 5 presents the experimental evaluation of our approach. Section 6 and 7 conclude the paper with related work and final conclusion.

2 MOTIVATION

In this section, we present an overview of how frequency scaling works on modern GPUs, and the key insights and challenges in applying frequency scaling to different applications and workloads. We present two speedup-energy characterization results performed on two applications running on an NVIDIA V100 GPU.

2.1 Saving Energy by Frequency Scaling

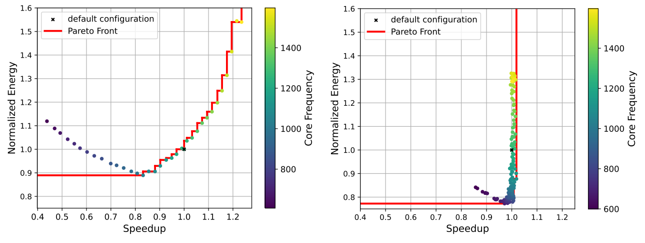
Dynamic Voltage Frequency Scaling (DVFS) allows for exploring new opportunities in optimizing applications' energy consumption. While DVFS is able to significantly reduce the energy consumption of a task, this typically comes at the cost of performance; therefore, the problem is a multi-objective one, where we can explore different tradeoffs. Previous work shows that this trade-off is not trivial, and good energy savings can be achieved at the cost of a negligible loss in performance [11]. As the search space of frequency configurations can be very large, the typical approach is to highlight the configurations that have a dominant energy-performance trade-off. These "optimal" frequencies can be obtained by computing the Pareto-optimal solutions. These solutions represent the boundary of the set of all possible outcomes where no improvement can be made in one objective without sacrificing the improvement of at least one other objective. Particularly, in our case the Pareto frontier would depict all the combinations of speedup and normalized energy where no other combination can achieve higher speedup without increasing the energy or reducing the energy without decreasing the speedup.

Heterogeneous systems are often programmed by using heterogeneous programming models such as OpenCL or SYCL, which provide code portability to different platforms. DVFS, however, is made available through different, vendor-specific, libraries such as NVML for NVIDIA GPUs or ROCm SMI for AMD GPUs. As our analysis is carried out on two SYCL-based applications, in this work we used the SYnergy API [1, 12], which interfaces with vendor-specific libraries from NVIDIA, AMD, and Intel GPUs and allows for portable energy profiling and frequency scaling.

2.2 Energy Saving depends on the Application

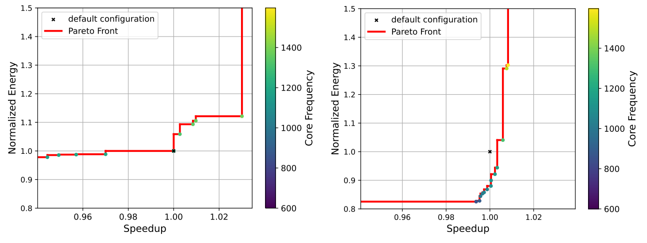
Frequency scaling can have a very different impact on energy depending on the type of application. For compute-bound applications, we can have performance improvement at the cost of higher energy consumption by increasing the core frequency. On the other hand, memory-bound applications may benefit from core down-scaling to reduce energy consumption with small performance degradation. Figure 1 shows how different core frequency configurations can affect the speedup and energy of two real-world applications LiGen (Figure 1a) and Cronos (Figure 1b). The baselines used for computing the speedup and normalized energy are the time and energy of the application executed with the default core frequency. The percentage improvement or loss in speedup and normalized energy due to frequency scaling are computed with respect to the default frequency configuration.

In LiGen the core frequency can be raised to produce speedup improvements of up to 25%. While, decreasing the core frequency leads to smaller energy savings, up to 10%, at the expense of a 15% loss in performance. For the Cronos application, by decreasing the core frequency, we can achieve energy savings up to 22% with a speedup loss close to 0%. While raising the core frequency increases the energy consumption by 30% without providing any significant improvement in performance. Due to the multi-objective nature



(a) LiGen (b) Cronos

Figure 1: LiGen and Cronos multi-objective characterization.



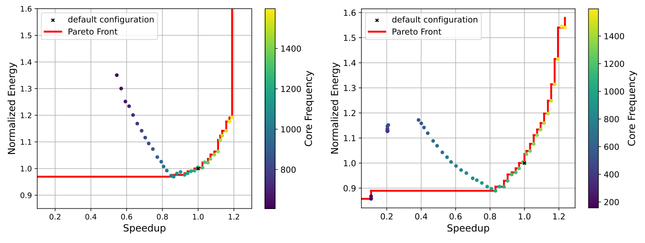
(a) Small input size (b) Large input size

Figure 3: Cronos multi-objective characterization with Pareto-optimal solution on input size 20x8x8 (a) and 160x64x64 (b).

of the problem, there is no single frequency that achieves at the same time the best performance and energy consumption. However, using the Pareto-set we can explore the solutions that can provide different trade-offs between speedup and energy. Through this analysis, it is possible to choose a Pareto-optimal frequency configuration according to the energy constraints of the specific use case.

2.3 Energy Saving depends on the Workload

Exploring the trade-off between energy and speedup can be even more challenging as the energy behavior of the same application can be affected by the workload size.



(a) Small input size (b) Large input size

Figure 2: LiGen multi-objective characterization with Pareto-optimal solutions on input size of 2 ligands x 89 atoms x 8 fragments (a) and 10000 ligands x 89 atmos x 20 fragments (b).

Figure 2 shows speedup and normalized energy consumption for the LiGen application with an input size of 2 ligands x 89 atoms x 8 fragments and 10000 ligands x 89 atoms x 20 fragments. The red line highlights the Pareto-optimal solutions.

In Figure 2a, we notice a speedup up to 20% by increasing the core frequency while consuming 20% more energy. Differently, decreasing the core frequency does not provide any energy savings. For large input size Figure 2b, the energy behavior is the opposite. By decreasing the core frequency, we can notice energy saving is up to 10% with a speedup loss of 10%, while the same performance improvement in Figure 2a comes with a 60% increase in energy consumption.

Figure 3 shows the Pareto-optimal solutions for the Cronos application with an input size of 20x8x8 and 160x64x64. For a small input size (Figure 3a) we notice a speedup up to 3% with a 10% growth in energy consumption by increasing the core frequency. Differently, decreasing the core frequency does not provide any significant energy savings. For larger input sizes (Figure 3b), decreasing the core frequency allows for significant energy saving up to 20%, while only losing 1% speedup. On the other hand, we have no speedup improvement, but an increase in energy consumption of up to 30% by increasing the core frequency.

Based on these observations, we build a more accurate domain-specific multi-objective model that seeks to automatically predict Pareto-optimal frequency configurations of a specific application based on the input characteristics.

3 ENERGY CHARACTERIZATION OF TWO REAL WORLD APPLICATIONS

In this section, we provide an energy characterization of a magnetohydrodynamics code (Cronos) and a drug discovery framework (LiGen) on a set of hardware composed of NVIDIA V100 and AMD MI100 GPUs. As defined in Section 2.2 the percentage improvement (or loss) in speedup and normalized energy are computed with respect to the default frequency configuration.

3.1 Magnetohydrodynamics

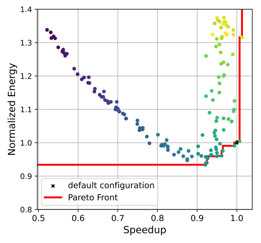
Cronos [28, 29, 38] is a magnetohydrodynamics (MHD) code developed for the solution of plasma-dynamical problems in astrophysics and space science. Algorithm 1 shows the structure of the Cronos code. The pseudocode outlines a simplified version of the basic structure of the Cronos application. The main computation happens in the function *computeChanges*, which computes the changes that occur for every element of the grid, as well as the CFL (Courant-Friedrichs-Lewy) value for every grid cell. This function represents a 13-point stencil application, where all cells can be computed in parallel. Due to the finite volume scheme involved, they need access to their neighborhood of 2 cells in each direction, i.e. 4 cells in each dimension. The next step in the algorithm is a reduction using the max operation on the CFL values of each cell. This step is parallelized using parallel reductions. After that the *integrateTime* function applies the previously computed changes to the grid, updating the state of every cell. This function is parallelized for every cell in the grid. The last step of the main computational loop is to update the boundary of the grid. This function only touches the outermost surfaces of the entire grid in parallel, rather than every cell like in the previous steps. In every timestep, the *timeDelta* is adjusted using the maximum *cfl* value which then advances the simulation by one timestep. The whole simulation runs until a preconfigured *endTime* is reached.

In our study, we explore how frequency scaling can affect the trade-off between energy and speedup as the input size increases.

Algorithm 1: Cronos algorithm

```
1  $grid[SIZE\_Z][SIZE\_Y][SIZE\_X]$;
2  $grid \leftarrow initialise()$;
3  $grid \leftarrow applyBoundary(grid)$;
4  while $currentTime \leq endTime$ do
5      for $substep \leftarrow 0$ to 2 do
6          $changeBuf[SIZE\_Z][SIZE\_Y][SIZE\_X]$;
7          $cflBuf[SIZE\_Z][SIZE\_Y][SIZE\_X]$;
8          $cflBuf, changeBuf \leftarrow computeChanges(grid)$;
9          $cfl \leftarrow reduce(cfl, cflBuf, max)$;
10         $grid \leftarrow integrateTime(grid, changeBuf, substep)$;
11         $grid \leftarrow applyBoundary(grid)$;
12     end
13     $timeDelta \leftarrow adjustTimestepDelta(timeDelta, cfl)$;
14     $currentTime$ += $timeDelta$;
15 end
```

3.1.1 *Energy scalability on grid dimensions.*


(a) 10*x*4*x*4 grid size

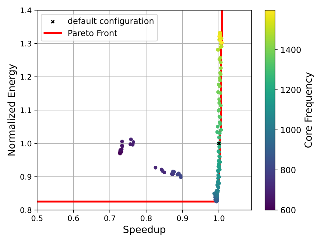
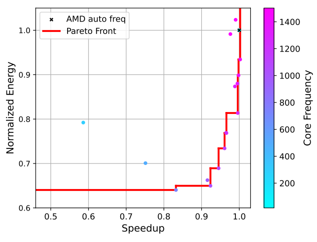

(b) 160*x*64*x*64 grid size

Figure 4: Cronos multi-objective characterization on NVIDIA V100 with small (10*x*4*x*4) and large (160*x*64*x*64) grid size.

Figure 4 and 5 show the speedup and normalized energy of the Cronos application executed on NVIDIA V100 and AMD MI100 GPUs with a small (10*x*4*x*4) and large (160*x*64*x*64) grid sizes. The red line highlights the Pareto-optimal solutions. On NVIDIA V100 GPU (Figure 4a and 4b) for both small and large grids, increasing the core frequency leads to higher energy consumption, up to 40%, with respect to the default configuration, without any improvement in performance. Energy-wise, as the grid size increases we can have a higher chance of energy saving with a loss of speedup close to 0% by lowering the core frequency.

Figure 5a and 5b show the results on the AMD MI100 GPU. AMD GPUs do not have a default frequency, but instead use a performance level for dynamic frequency change. Normally the


(a) 10*x*4*x*4 grid size

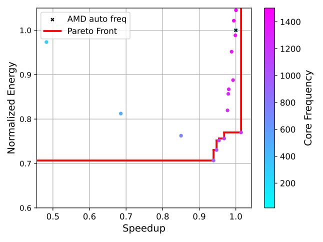

(b) 160*x*64*x*64 grid size

Figure 5: Cronos multi-objective characterization on AMD MI100 with small (10*x*4*x*4) and large (160*x*64*x*64) grid size.

performance level is set to “automatic”, which is the configuration that we consider as the default behavior of the GPU. We can see that the default setting is very close to the higher achievable speedup, but energy consumption can be reduced by lowering the frequency. In particular, for small grid sizes, the achievable energy saving is around 35% with a speedup loss of about 10%. On the other hand, larger grid sizes have lower energy savings, with a difference of about 5%, while the speedup loss remains the same as the small input size.

3.2 Drug Discovery

LiGen is the molecular docking engine part of the EXSCALATE [14] virtual screening platforms. It aims at finding the best drug candidates to test *in-vitro* and *in-vivo* in a drug discovery process. In this context, we have a target protein representing the disease and a very large chemical library of ligands (small molecules) representing the possible drugs. The platform’s goal is to rank the chemical library according to the ligand-protein interaction strength to identify the best candidates to forward to the next stages of drug discovery. Algorithm 2 shows the two main tasks that allow to compute the interaction strength. The first one is named *dock*, and it aims at estimating the 3D displacement of the ligand’s atoms when it interacts with the protein, i.e., dock the ligand inside the protein (lines 2-12). The second task is named *score*, and it aims at computing the interaction strength of the ligand-protein pair (lines 13-18). LiGen is the application that carries out these tasks. It provides the computation kernel for different devices with the C++*17*, CUDA, and SYCL implementations.

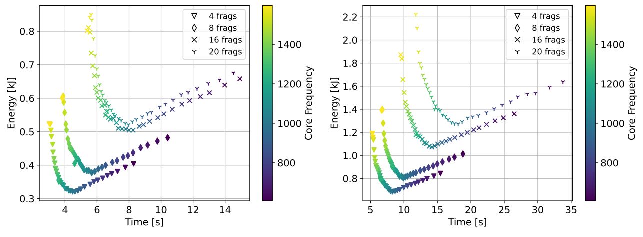
All the ligand-protein evaluations are independent. Thus, the problem is embarrassing parallel. Moreover, the protein is a constant for each virtual screening campaign. Thus, we can analyze the algorithm complexity using features of the ligands. From the asymptotic complexity analysis [14, 42], the complexity of the single ligand-protein evaluation depends on the ligand’s number of atoms and the number of rotamers. The latter is a subset of the ligand’s bonds that LiGen can use to change its geometric shape without altering physical and chemical properties. In particular, each rotamer splits the ligand’s atoms into two disjoint sets that can rotate independently along the rotamer axis. In this document, we refer to each set as a ligand *fragment*.

In this section, we provide a multi-objective analysis highlighting how frequency scaling can affect LiGen’s energy consumption and performance when given different numbers and types of ligands as inputs.

Algorithm 2: LiGen virtual screening algorithm

```
Data: num_restart, num_iterations, max_num_poses
Input: ligand, target
Output: score
1  $scores \leftarrow \emptyset$, $poses \leftarrow \emptyset$;
2  for $i \leftarrow 0$ to $num\_restart$ do
3      $pose \leftarrow initialize\_pose(ligand, i)$;
4      $pose \leftarrow align(pose, target)$;
5      for $n \leftarrow 0$ to $num\_iterations$ do
6          for $fragment \leftarrow pose.fragments$ do
7              $pose \leftarrow optimize(pose, fragment, target)$;
8          end
9      end
10     $pose \leftarrow evaluate(pose, target)$;
11     $poses \leftarrow poses \cup pose$;
12 end
13 $poses \leftarrow clip(sort(poses), max\_num\_poses)$;
14 for $pose \leftarrow poses$ do
15     $score \leftarrow compute\_score(pose, target)$;
16     $scores \leftarrow scores \cup score$;
17 end
18 return $max(scores)$
```

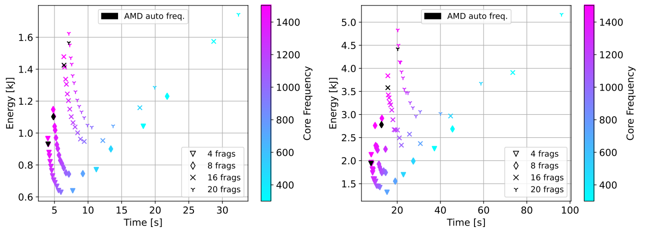
3.2.1 *Energy scalability on atoms and fragments.*



(a) 31 atoms (b) 89 atoms

Figure 6: LiGen multi-objective characterization on NVIDIA V100 scaling the number of fragments with a fixed number of atoms.

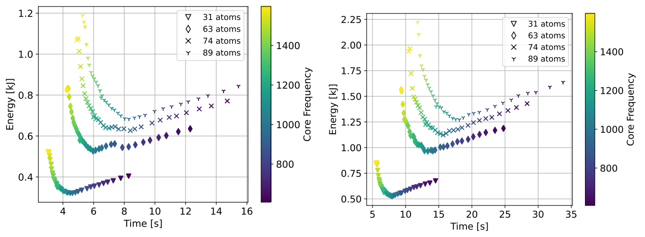
For this analysis, we use raw values instead of normalized values to avoid an overlap of the energy curves and have a better visualization of the data as the input size increases. Figure 6 shows the energy and time behavior of the LiGen application running on NVIDIA V100 GPU with 100000 ligands. Each ligand is composed of a small (Figure 6a) and a large number of atoms (Figure 6b) while varying the number of fragments. As the number of fragments increases, both the energy consumption and time increase. In particular, this behavior is more evident with a large number of atoms.



(a) 31 atoms (b) 89 atoms

Figure 7: LiGen multi-objective characterization on AMD MI100 scaling the number of fragments (4, 8, 16, 20) with a fixed atom size.

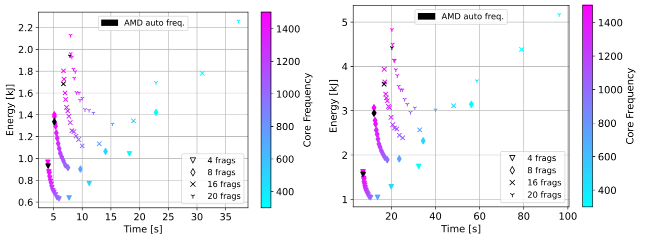
Figure 7 shows the results using the same experiments performed in Figure 6 on an AMD MI100 GPU. As with the NVIDIA V100 GPU, increasing the number of fragments produces an increase in energy consumption and time. Additionally, on AMD MI100 we can observe greater energy consumption variations with a large atom size, as the number of fragments increases. Compared with the NVIDIA V100 results, both energy and time are higher on AMD MI100.



(a) 4 fragments (b) 20 fragments

Figure 8: LiGen multi-objective characterization on NVIDIA V100 scaling the number of atoms (31, 63, 74, 89) with a fixed fragment size.

In Figure 8 we explore the energy and time behavior of LiGen with a fixed number of fragments while increasing the number of atoms in the ligands (Figure 8a and Figure 8b). By only increasing the number of atoms, we can notice more variability in energy consumption with respect to Figure 6.



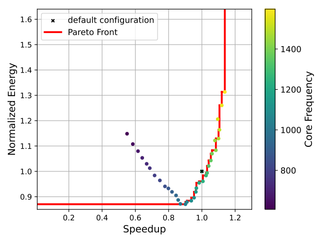
(a) 4 fragments (b) 20 fragments

Figure 9: LiGen multi-objective characterization on AMD MI100 scaling the number of atoms (31, 63, 74, 89) with a fixed fragment size.

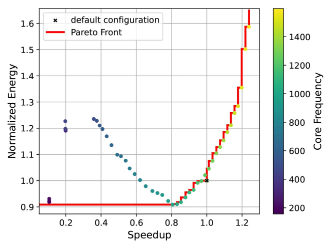
The same experiments with a fixed fragment size have been reproduced on the AMD MI100 GPU, showing a similar behavior that can be seen in Figure 9.

3.2.2 *Energy scalability on number of ligands.*

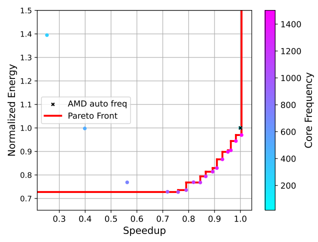
All the computation performed during the docking and scoring phases of LiGen can be easily parallelized by allowing each kernel on the GPU to compute several ligands simultaneously. By scaling the number of ligands computed in a kernel, we may have a different utilization of the GPU resources that may affect the energy behavior of the whole application.



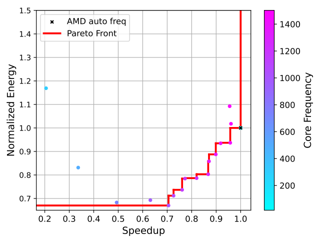
(a) Small input NVIDIA V100



(b) Large input NVIDIA V100



(c) Small input AMD MI100



(d) Large input AMD MI100

Figure 10: LiGen multi-objective characterization on NVIDIA V100 and AMD MI100 with small (256 ligands x 31 atoms x 4 frag.) and large (10000 ligands x 89 atoms x 20 frag.) input size.

Figure 10 shows the speedup and normalized energy with the Pareto-optimal solutions of the LiGen application on a set of small and large ligands. On NVIDIA V100 (Figure 10a and Figure 10b) we may have a speedup up to 22% with an energy consumption increase of 60% by using a large input size. Differently, on small input sizes, the achieved speedup is lower, up to 15%, while the energy consumed is 30% compared to the 60% on large input sizes. Furthermore, on small input, we have more chance of saving energy. Looking at the Pareto-optimal solutions on the red line we can select some frequency configurations that provide a 10% of energy saving with a speedup loss of 5%. For the AMD MI100 GPU (Figure 10c and Figure 10d), the normalized energy and speedup are computed with respect to the frequency automatically selected by the GPU. This frequency always performs better on both small and large inputs. The Pareto-optimal solutions highlight an energy behavior similar to the results on NVIDIA V100. With small input sizes, we can select Pareto-optimal frequency configurations that achieve 20% energy saving with a speedup loss of 10%. While for large inputs the same energy saving comes at the cost of higher speedup loss.

4 DOMAIN-SPECIFIC ENERGY MODELING

4.1 General-purpose Energy Modeling

To optimize the energy efficiency brought by GPU DVFS, many existing models generally adopted learning-based methods to identify kernel patterns and estimate the effects of DVFS on energy consumption. In particular, Fan *et al.* [11] proposed a machine-learning model to optimize energy consumption statically by taking DVFS into account on GPUs. The model is based on typical two-phase modeling with supervised learning: training phase and prediction phase. In the training phase, Fan *et al.* first build 106 carefully-designed micro-benchmarks and extract a set of static features of each micro-benchmark. Successively, each micro-benchmark is executed with various frequency configurations to obtain energy measurements. Those measurements, together with frequency configurations and static features, are used to train the normalized energy consumption model. All the micro-benchmarks are built to stress one or more features that characterize the device's energy consumption and they can be used to train a general-purpose model that works on different applications. In the prediction phase, static code features are extracted from a new input code. Those features, combined with frequency configurations and the previously trained models, are used to predict the normalized energy consumption of a new code.

The approach, proposed by Fan *et al.*, takes advantage of static code features to predict the normalized energy consumption of a new code without executing it. However, the static code features have more weight on computing ability, which leads to a higher prediction accuracy of compute-bound applications and lower prediction accuracy of memory-bound applications.

4.2 Towards Domain-specific Modeling

As demonstrated in Section 3 the energy behavior of real-world applications may depend on the input characteristics. State-of-the-art general-purpose models only consider static features without taking into account how the application can be affected by different workloads and input types. To improve the accuracy of general-purpose models we provide two domain-specific energy models that rely on the input characteristics of a program to predict the speedup and normalized energy for each device core frequencies.

The methodology used for modeling is based on supervised learning, in which each model is built during the training phase using domain-specific features. Then, the prediction phase takes unseen features as input and generates execution time and energy consumption, which are used to compute speedup and normalized energy for each frequency configuration of the target hardware. Table 1 shows the features used in the general-purpose model.

4.2.1 *Features selection.*

The domain-specific nature of the models implies that there is not a said set of features for every application, but rather, for each application different features must be chosen through an in-depth time and energy analysis. In our case, the Cronos and LiGen models are built on top of the energy characterization analysis in Section 3, which provides interesting insights to select the features that best represent the energy characterization of the target application.

Table 1: General-purpose model features.

Feature	Description
f_{int_add}	integer additions and subtractions
f_{int_mul}	integer multiplications
f_{int_div}	integer divisions
f_{int_bw}	integer bitwise operations
f_{float_add}	floating point additions and subtractions
f_{float_mul}	floating point multiplications
f_{float_div}	floating point divisions
f_{sf}	special functions
f_{gl_access}	global memory accesses
f_{loc_access}	local memory accesses

Magnetohydrodynamics. The characterization in Figure 4 and 5 shows that the Cronos application exhibits different behaviors when considering varying grid sizes. For this reason, we model the behavior of the application through the use of the grid size on the x,y, and z-axis.

Drug discovery. As expected, the energy consumed by the LiGen application increases as the number of ligands increases (Figure 10). Furthermore, Figure 6 and 8 show that the LiGen energy and time are strictly related to the structure of the ligands, e.g., the number of fragments and atoms in each ligand. To fully capture all these aspects, our domain-specific models use the number of ligands, fragments, and atoms as features.

The final features for the two applications are summarized in Table 2.

Table 2: Domain-specific model features.

Application	Features
Cronos	$f_{grid\_x}, f_{grid\_y}, f_{grid\_z}$
LiGen	$f_{ligands}, f_{fragments}, f_{atoms}$

4.2.2 *Training phase.*

The workflow of the training phase is illustrated in Figure 11. For each application, the outputs of this phase are two models: one for execution time and one for energy consumption. In order to build the training set, the models require both the input features $\vec{f}$ and the ground truth values for the execution time (t) and energy consumption (e) of the application. Furthermore, as we must capture the behavior of the application (1) with different frequencies, each input must be executed for each (or a part) of the frequency configuration (c) of the target hardware (2). Once the training dataset $D = \{s : s = (\vec{f}, c, t, e)\}$ is built (3), we apply different machine learning algorithms to build the two models $T(\vec{f}, c)$ for execution time (4) and $E(\vec{f}, c)$ for energy consumption (5).

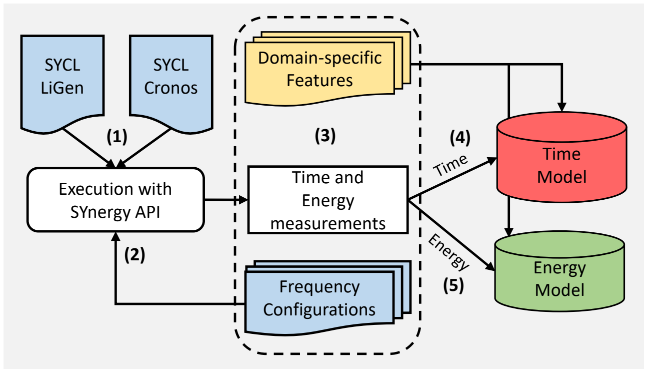


Figure 11: Domain-specific model training phase

4.2.3 *Prediction phase.*

The workflow of the prediction phase is presented in Figure 12. During the prediction phase, given a new features vector $\vec{f}'$ (1) and a frequency configuration c' (2), the models are used to predict $\hat{t} = T(\vec{f}', c')$ and $\hat{e} = E(\vec{f}', c')$. Once the energy and time predictions for all frequency configurations are available, the predicted speedup (3) and normalized energy (4) can be computed, taking as a baseline the predicted values for the default frequency configuration. Finally, speedup and normalized energy are used to select the Pareto-optimal solutions.

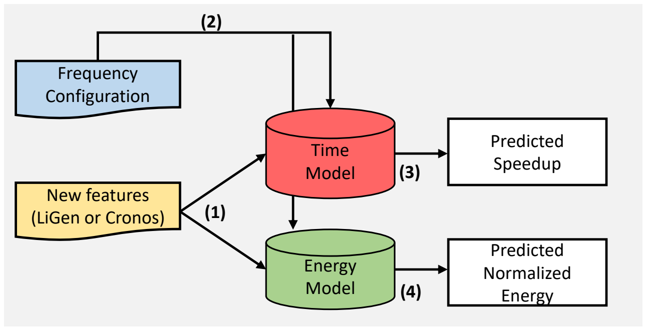


Figure 12: Domain-specific model prediction phase

5 EXPERIMENTAL EVALUATION

In this section, we present the experimental evaluation of the proposed models, along with a comparison with the state-of-the-art model that is based on static code features.

5.1 Experimental Setup

The models' training dataset is obtained by launching the application on a system using Ubuntu 22.04, an Intel Xeon Gold 5218 CPU, and an NVIDIA V100 GPU with 32 GB of High Bandwidth Memory (HBM2). The NVIDIA driver version and CUDA version are respectively 530.30.02 and 12.1. The V100 GPU supports one memory frequency (1107 MHz) and 196 core frequencies from 135 MHz to 1597 MHz. The execution time is profiled through the standard C++ library, while the energy consumption is profiled through the SYnergy API. Each experiment is repeated five times to reduce the impact of any outliers. We launched the two applications, Cronos and LiGen, with many different inputs. For Cronos, we use five grid configurations, starting from a $10x4x4$ grid up to a $160x64x64$ grid. LiGen experiments are executed on different numbers of ligands that have varying numbers of fragments and atoms. Each experiment can be represented by a tuple $(l, a, f) \in \{2, 16, 1024, 4096, 10000\} \times \{31, 63, 71, 89\} \times \{4, 8, 16, 20\}$, where l is the number of ligands, a is the number of atoms and f is the number of fragments. To validate our approach the applications were executed with all the frequencies of the V100 GPU and then we highlighted the point predicted by the general-purpose model and domain-specific model.

5.2 Domain-specific versus General-purpose Models Accuracy

The general-purpose models are built using a set of well-defined micro-benchmarks that stress different architectural components of the target hardware [11, 20]. Then, the prediction phase for general-purpose models uses the static code features of the application to predict the energy and execution time values.

Differently, the presented domain-specific models are trained on a specific application and are meant to be used for that same application. Thus, we validate these models by using leave-one-out cross-validation over the domain-specific features dataset as defined in Table 2. Specifically, for each different input feature $\vec{f}$ we build a set D_v for validation and a set D_t for training, defined as follows:

$$D_v = \{s \in D : s \text{ has input features } \vec{f}\}$$
$$D_t = D \setminus D_v$$

where $s = (\vec{f}, c, t, e)$ as defined in Section 4.2.2.

5.2.1 *Accuracy of Speedup and Normalized Energy predictions.*

Our approach is built on top of two models: one is to predict speedup and the other is to predict normalized energy. Through the *scikit-learn* Python library, we perform the training phase of these models using different kinds of regression algorithms (Linear, Lasso, SVR_RBF, Random Forest) and finally select the algorithm that achieves the highest accuracy for each model. Random Forest achieves the maximum accuracy for both normalized energy and speedup models. We performed a hyperparameter tuning of the Random Forest regression algorithm through a grid search method. The grid space has been defined by three parameters: the maximum depth of the tree (max_depth); the number of trees in the forest (n_estimators); the number of features to consider when looking for the best split (max_features). As a result, the default parameter performs better for both the speedup and energy models. The accuracy comparison is based on the analysis of the mean absolute percentage error (MAPE) of the models. For each application, we first compute the absolute percentage error between the predicted values by the two models and the true values obtained through running the application with each frequency configuration. Then, we measure the prediction accuracy as the mean of the absolute percentage error over all the frequency configurations. Figure 13 shows the speedup and normalized energy prediction accuracy of the two models expressed as MAPE for both applications Cronos and LiGen on different input workloads. The domain-specific models achieve a lower error for both applications on every input and are at least 10 times more accurate than the general-purpose model.

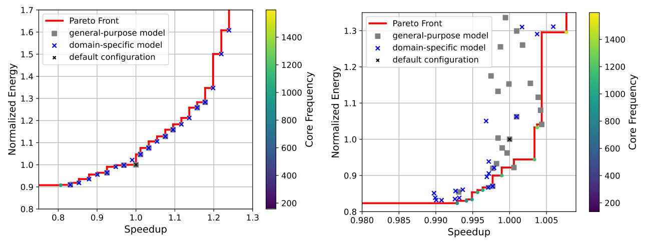
5.2.2 *Accuracy of Predicted Pareto Set.*

The final purpose of the models is to predict Pareto-optimal frequency configurations that allow for energy saving or a boost in performance. In the accuracy analysis, the general-purpose and domain-specific models are both used to predict the execution time and the energy consumption for each frequency configuration. Then to obtain the Pareto-optimal frequency configurations, we

(1) compute the predicted speedup and normalized energy, using the predicted values of the default frequency configuration as baseline;
(2) compute the predicted Pareto-optimal solutions;
(3) create the Pareto-optimal frequency configuration set by selecting the frequency configurations associated with each predicted Pareto-optimal solution.

This process produces two sets of predicted Pareto-optimal frequency configurations, one for the general-purpose models and one for the domain-specific ones, that are comparable with the actual Pareto-optimal frequency configuration.

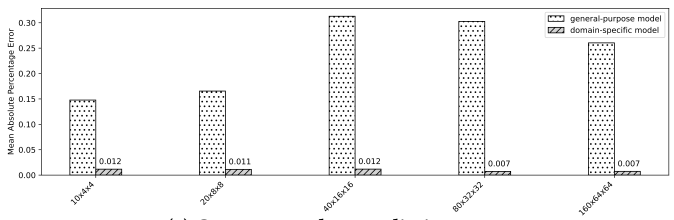
To assess the models' accuracy we use the speedup and normalized energy obtained running the applications with the predicted Pareto-optimal frequency configurations. In fact, these are the real values that would be obtained if the applications were executed with the predicted Pareto-optimal frequencies. Analyzing the accuracy of the predicted Pareto-optimal solutions is not trivial as the actual points obtained by the predicted Pareto-optimal frequency configurations are not for sure Pareto-optimal. In general, a better Pareto approximation is a set of solutions that, in terms of speedup and normalized energy, is the closest to the real Pareto-optimal one. Moreover, the number of predicted Pareto-optimal frequency configurations that exactly match the true Pareto-optimal frequency configurations can be considered as a metric when comparing accuracy between models.



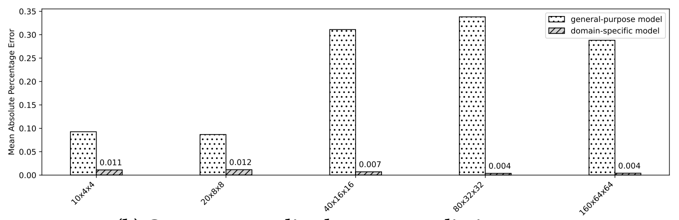
(a) LiGen ($10000x89x20$) **(b) Cronos ($160x64x64$)**

Figure 14: LiGen and Cronos Pareto-optimal solution predicted by the general-purpose and domain-specific models compared with the true Pareto-set (red line)

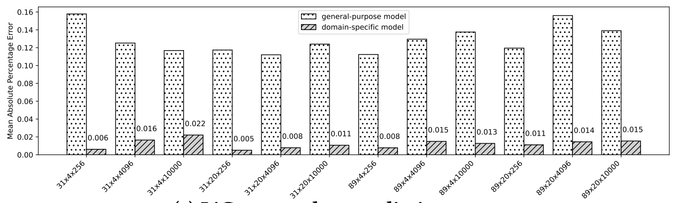
Figure 14 shows the predicted Pareto-optimal solutions for the LiGen and Cronos on the same input of Figure 10b and 4b. The red line represents the true Pareto-optimal configurations, while the gray squares and the blue crosses represent respectively the



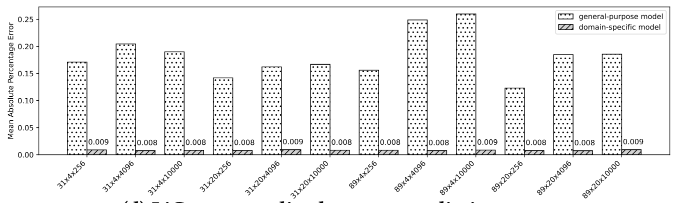
(a) Cronos speedup prediction error



(b) Cronos normalized energy prediction error



(c) LiGen speedup prediction error



(d) LiGen normalized energy prediction error

Figure 13: Comparison of prediction accuracy for Cronos and LiGen using the general-purpose and domain-specific models.

Pareto-optimal solutions predicted by the general-purpose and the domain-specific models. The predicted frequency configurations of the general-purpose model for LiGen fully match the true Pareto set, while the domain-specific model predicts a configuration that is not in the true Pareto set. On the other hand, the domain-specific model predicts more Pareto-optimal points than the general-purpose model. This is a positive trait of the model, as it allows us to explore deeply the trade-off between speedup and normalized energy. For example, in Figure 14a only the domain-specific model is able to predict the highest-performing frequencies achieving around 23% of speedup compared to the 18% speedup obtained by using the general-purpose model prediction. For the Cronos application, the general-purpose and the domain-specific models have more difficulty in predicting the speedup as the values are very close to each other. Therefore, differently from LiGen, both models predict fewer frequency configurations that exactly match the true Pareto-optimal configurations. With respect to energy, the domain-specific model is more precise in the prediction of frequency configurations that correspond to normalized energy values in line with the true Pareto-optimal solutions.

6 RELATED WORK

The energy sustainability of modern HPC systems is a critical concern from both an economic and an ecological standpoint. For this reason, many efforts in the scientific community are directed toward the development of tools that can contribute to reducing the environmental footprint of HPC systems. In particular, in recent years, many studies on modeling the energy consumption of scientific applications running on large-scale clusters have been published, enabling researchers to identify new strategies to reduce their energy consumption [6, 15, 20, 21, 30, 43]. Among them, Lopes *et al.* [30] proposed a model that relies on extensive GPU micro-benchmarking using a cache-aware roofline model. Wu *et al.* [43] studied the performance and energy models of an AMD GPU by using K-means clustering. Guerreiro *et al.* [20] made more improvements: they not only presented the approach of gathering performance events by micro-benchmarks in detail but also predicted how the GPU voltage scales.

Such methods are often based on power limitation or frequency modulation of computing systems. In terms of power limitation, Remesh *et al.* [36] modeled the impact of dynamic power capping schemes in progress for a set of online HPC applications. Hao *et al.* [23] combined the powercap with uncore frequency scaling and proposed a machine learning modeling to predict the Pareto-optimal powercap configurations for achieving trade-offs among performance and energy consumption. In terms of frequency modulation, Ge *et al.* [16] applied fine-grained GPU core frequency and coarse-grained GPU memory frequency on a Kepler K20c GPU, but only analyzed three compute-intensive benchmarks to study the impact of GPU DVFS.

However, additional challenges and opportunities arise when these tools must be integrated into existing systems using workload managers [22, 39, 45, 46]. Furthermore, most of the above energy models are proposed for general purposes, which may lead to lower prediction accuracy of energy consumption targeting specific applications. Our proposal, starting from those challenges, specifically targets drug discovery and magneto-hydrodynamics applications and outperforms the general-purpose energy model.

Other works propose tools to dynamically modulate frequency or adjust the job's power cap by analyzing features of the application at run-time [7, 10]. These tools lack an easily portable approach, either because designed for specific hardware or because they require a custom implementation in order to support different kinds of architectures. Differently, our approach provides a model-based and architecture-independent solution that only requires the range of frequency configuration to work on any architecture. Also, this solution can be easily integrated into other existing toolchains to drive the frequency selection of Pareto-optimal solutions.

Drug discovery in HPC. How to dock a ligand inside a protein and how to estimate their interaction strength are well-known problems in the literature that we need to solve for different purposes [5, 41]. Indeed, we can find a wide range of software that can dock and score, covering the accuracy-performance spectrum [4, 32, 35, 44]. In this work, we focus on virtual screening, the initial stage of drug discovery that suggests which molecules to test *in-vitro* and *in-vivo*. Recent studies demonstrated how the introduction of this stage increases the success probability of the drug discovery pipeline [17]. Since this problem is embarrassingly parallel and the chemical space is huge, the size of the chemical library that we need to virtual screen is constrained only by the available computation effort. Recently, the largest campaigns of virtual screening performed billions [18] and a trillion [14] of protein-ligand evaluations against SARS-CoV-2. The latter used LiGen to carry out the dock and score computation, using almost all the nodes of two supercomputers (HPC5 at ENI and MARCONI100 at CINECA).

Cronos code for Astrophysical Magnetohydrodynamics. The use of numerical methods is common in the field of astrophysics, to complement real behaviors observed in experiments with simulations. There is a wide range of applications developed for solving problems in hydrodynamics or magnetohydrodynamics, such as Racoon [9], Ramses [13], Nirvana [47], Amrvac [27, 40], Athena [37], Pluto [31], WENO–WOMBAT [8], each developed with a particular problem focus. The Cronos code [28, 29], instead, was developed so that it could easily adapt to the various problems investigated in the field of astrophysical modeling. In addition, the code also allows the solver to be used for other conservation laws that can be provided by the user. Recently, the Cronos code related to the solver has been ported to the SYCL [19] and Celerity [38] programming models to run on a single node and a distributed memory cluster, respectively.

7 CONCLUSION AND FUTURE WORKS

To address the energy sustainability challenges, we build domain-specific energy models capable of predicting Pareto-optimal frequency configurations by combining frequency scaling with a machine-learning approach. As a preliminary investigation to build domain-specific models, we perform an energy evaluation of two real-world applications, LiGen and Cronos, on an AMD MI100 and an NVIDIA V100, demonstrating how differing workloads can significantly impact the energy behavior of both applications. Based on it, we provide a methodology to build domain-specific models capable of predicting the Pareto-optimal frequency configurations by leveraging the input characteristics of the target application instead of static code features. Finally, we validate the accuracy of our methodology by comparing the speedup and normalized energy prediction of the LiGen and Cronos models with the general purpose model and the actual values. In addition, we provide a more detailed analysis based on the comparison of Pareto optimal solutions. The results show that domain-specific models have at least a ten times lower error than the general-purpose approach. Moreover, the Pareto set comparison highlights that the domain-specific approach allows a better exploration of the speedup-energy trade-off, as it predicts more solutions on the true Pareto set. As future work these models can be easily integrated into the SYnergy compilation toolchain [12] by replacing the general-purpose SYnergy model with our domain-specific model. In this way, we can use the energy target metric defined in SYnergy to select a specific frequency configuration that fits the defined energy target. Additionally, using SYnergy's support for per-kernel frequency scaling, we can use the domain-specific model to select a different frequency configuration for each kernel of the application by focusing on each kernel's input rather than the input for the entire program.

ACKNOWLEDGMENTS

This research has been funded by the European High-Performance Computing Joint Undertaking (JU) under grant agreement No. 956137 (LIGATE project) and No. 956560 (REGALE project).

REFERENCES

[1] 2023. SYnergy API. https://github.com/unisa-hpc/SYnergy

[2] AMD. 2023. ROCm System Management Interface. https://github.com/RadeonOpenCompute/rocm_smi_lib

[3] Andrea R Beccari, Carlo Cavazzoni, Claudia Beato, and Gabriele Costantino. 2013. LiGen: a high performance workflow for chemistry driven de novo design.

[4] Jacek Biesiada, Aleksey Porollo, Prakash Velayutham, Michal Kouril, and Jaroslaw Meller. 2011. Survey of public domain software for docking simulations and virtual screening. *Human Genomics* 5, 5 (2011), 497–505.

[5] Alexander V. Boukhanovsky, Valeria V. Krzhizhanovskaya, and Marian Bubak. 2018. Urgent computing for decision support in critical situations. *Future Generation Computer Systems* 79 (2018), 111–113. https://doi.org/10.1016/j.future.2017.11.003

[6] Jee W. Choi and Richard W. Vuduc. 2016. Analyzing the Energy Efficiency of the Fast Multipole Method Using a DVFS-Aware Energy Model. In *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. 79–88. https://doi.org/10.1109/IPDPSW.2016.206

[7] Julita Corbalan, Oriol Vidal, Lluis Alonso, and Jordi Aneas. 2021. Explicit uncore frequency scaling for energy optimisation policies with EAR in Intel architectures. In *2021 IEEE International Conference on Cluster Computing (CLUSTER)*. 572–581. https://doi.org/10.1109/Cluster48925.2021.00089

[8] JMF Donnert, H Jang, P Mendygral, Gianfranco Brunetti, D Ryu, and TW Jones. 2019. WENO–WOMBAT: Scalable Fifth-order Constrained-transport Magnetohydrodynamics for Astrophysical Applications. *The Astrophysical Journal Supplement Series* 241, 2 (2019), 23.

[9] Jürgen Dreher and Rainer Grauer. 2005. Racoon: A parallel mesh-adaptive framework for hyperbolic conservation laws. *Parallel Comput.* 31, 8-9 (2005), 913–932.

[10] Jonathan Eastep, Steve Sylvester, Christopher Cantalupo, Brad Geltz, Federico Ardanaz, Asma Al-Rawi, Kelly Livingston, Fuat Keceli, Matthias Maiterth, and Siddhartha Jana. 2017. Global extensible open power manager: A vehicle for HPC community collaboration on co-designed energy management solutions. In *High Performance Computing: 32nd International Conference, ISC High Performance 2017, Frankfurt, Germany, June 18–22, 2017, Proceedings 32*. Springer, 394–412.

[11] Kaijie Fan, Biagio Cosenza, and Ben H. H. Juurlink. 2019. Predictable GPUs Frequency Scaling for Energy and Performance. In *Proceedings of the 48th International Conference on Parallel Processing, ICPP, Kyoto, Japan, August 05-08*. 52:1–52:10.

[12] Kaijie Fan, Marco D'Antonio, Lorenzo Carpentieri, Biagio Cosenza, Federico Ficarelli, and Daniele Cesarini. 2023. SYnergy: Fine-grained Energy-Efficient Heterogeneous Computing for Scalable Energy Saving. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis SC'23*. https://doi.org/10.1145/3581784.3607055

[13] Sébastien Fromang, Patrick Hennebelle, and Romain Teyssier. 2006. A high order Godunov scheme with constrained transport and adaptive mesh refinement for astrophysical magnetohydrodynamics. *Astronomy & Astrophysics* 457, 2 (2006), 371–384.

[14] Davide Gadioli, Emanuele Vitali, Federico Ficarelli, Chiara Latini, Candida Manelfi, Carmine Talarico, Cristina Silvano, Carlo Cavazzoni, Gianluca Palermo, and Andrea Rosario Beccari. 2023. EXSCALATE: An Extreme-Scale Virtual Screening Platform for Drug Discovery Targeting Polypharmacology to Fight SARS-CoV-2. *IEEE Transactions on Emerging Topics in Computing* 11, 1 (2023), 170–181. https://doi.org/10.1109/TETC.2022.3187134

[15] Rong Ge, Xizhou Feng, and Kirk W Cameron. 2009. Modeling and evaluating energy-performance efficiency of parallel processing on multicore based power aware systems. In *2009 ieee international symposium on parallel & distributed processing*. IEEE, 1–8.

[16] Rong Ge, Ryan Vogt, Jahangir Majumder, Arif Alam, Martin Burtscher, and Ziliang Zong. 2013. Effects of Dynamic Voltage and Frequency Scaling on a K20 GPU. In *2013 42nd International Conference on Parallel Processing*. 826–833. https://doi.org/10.1109/ICPP.2013.98

[17] Jens Glaser, Josh V Vermaas, David M Rogers, Jeff Larkin, Scott LeGrand, Swen Boehm, Matthew B Baker, Aaron Scheinberg, Andreas F Tillack, Mathialakan Thavappiragasam, Ada Sedova, and Oscar Hernandez. 2021. High-throughput virtual laboratory for drug discovery using massive datasets. *The International Journal of High Performance Computing Applications* 35, 5 (2021), 452–468.

[18] Jens Glaser, Josh V Vermaas, David M Rogers, Jeff Larkin, Scott LeGrand, Swen Boehm, Matthew B Baker, Aaron Scheinberg, Andreas F Tillack, Mathialakan Thavappiragasam, Ada Sedova, and Oscar Hernandez. 2021. High-throughput virtual laboratory for drug discovery using massive datasets. *The International Journal of High Performance Computing Applications* 35, 5 (2021), 452–468. https:

//doi.org/10.1177/10943420211001565

[19] Khronos SYCL Working Group. 2021. *SYCL 2020 Specification, revision 3.* Technical Report. Khronos Group.

[20] Joao Guerreiro, Aleksandar Ilic, Nuno Roma, and Pedro Tomas. 2018. GPGPU Power Modeling for Multi-domain Voltage-Frequency Scaling. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA).* 789–800. https://doi.org/10.1109/HPCA.2018.00072

[21] João Guerreiro, Aleksandar Ilic, Nuno Roma, and Pedro Tomás. 2019. GPU Static Modeling Using PTX and Deep Structured Learning. *IEEE Access* 7 (2019), 159150–159161. https://doi.org/10.1109/ACCESS.2019.2951218

[22] Tomoaki Hamano, Toshio Endo, and Satoshi Matsuoka. 2009. Power-aware dynamic task scheduling for heterogeneous accelerated clusters. In *2009 IEEE International Symposium on Parallel & Distributed Processing.* 1–8. https://doi.org/10.1109/IPDPS.2009.5160977

[23] Meng Hao, Weizhe Zhang, Yiming Wang, Gangzhao Lu, Farui Wang, and Athanasios V. Vasilakos. 2021. Fine-Grained Powercap Allocation for Power-Constrained Systems Based on Multi-Objective Machine Learning. *IEEE Transactions on Parallel and Distributed Systems* 32, 7 (2021), 1789–1801. https://doi.org/10.1109/TPDS.2020.3045983

[24] Intel. 2023. Intel Level Zero API Specification. https://spec.oneapi.io/level-zero/latest/index.html

[25] Shailendra Jain, Surhud Khare, Satish Yada, V. Ambili, Praveen Salihundam, Shiva Ramani, Sriram Muthukumar, M. Srinivasan, Arun Kumar, Shasi Kumar, Rajaraman Ramanarayanan, Vasantha Erraguntla, Jason Howard, Sriram R. Vangal, Saurabh Dighe, Gregory Ruhl, Paolo A. Aseron, Howard Wilson, Nitin Borkar, Vivek De, and Shekhar Borkar. 2012. A 280mV-to-1.2V wide-operating-range IA-32 processor in 32nm CMOS. In *IEEE International Solid-State Circuits Conference, ISSCC.* 66–68.

[26] Ulya R. Karpuzcu, Nam Sung Kim, and Josep Torrellas. 2013. Coping with Parametric Variation at Near-Threshold Voltages. *IEEE Micro* 33, 4 (2013), 6–14. https://doi.org/10.1109/MM.2013.71

[27] Rony Keppens, Zakaria Meliani, Allard Jan van Marle, Peter Delmont, Alkis Vlasis, and Bart van der Holst. 2012. Parallel, grid-adaptive approaches for relativistic hydro and magnetohydrodynamics. *J. Comput. Phys.* 231, 3 (2012), 718–744.

[28] Ralf Kissmann, David Huber, and Philipp Gschwandtner. 2023. High-Resolution Simulations of LS 5039. *A&A (Forthcoming)* (2023). https://doi.org/10.1051/0004-6361/202345934

[29] Ralf Kissmann, Jens Kleimann, Barbara L. Krebl, and Tobias Wiengarten. 2018. The CRONOS code for astrophysical magnetohydrodynamics. *The Astrophysical Journal Supplement Series* 236, 2 (2018), 53.

[30] André Lopes, Frederico Pratas, Leonel Sousa, and Aleksandar Ilic. 2017. Exploring GPU performance, power and energy-efficiency bounds with Cache-aware Roofline Modeling. In *2017 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS).* 259–268. https://doi.org/10.1109/ISPASS.2017.7975297

[31] Andrea Mignone, C Zanni, Petros Tzeferacos, B Van Straalen, P Colella, and G Bodo. 2011. The PLUTO code for adaptive mesh computations in astrophysical fluid dynamics. *The Astrophysical Journal Supplement Series* 198, 1 (2011), 7.

[32] Natarajan Arul Murugan, Artur Podobas, Davide Gadioli, Emanuele Vitali, Gianluca Palermo, and Stefano Markidis. 2022. A Review on Parallel Virtual Screening Softwares for High-Performance Computers. *Pharmaceuticals* 15, 1 (2022), 63.

[33] United Nations. 2023. The 2030 Agenda for Sustainable Development. https://sdgs.un.org/2030agenda

[34] NVIDIA. 2023. NVIDIA NVML API Reference Guide. https://docs.nvidia.com/deploy/nvml-api/index.html

[35] Nataraj S. Pagadala, Khajamohiddin Syed, and Jack Tuszynski. 2017. Software for molecular docking: a review. *Biophysical Reviews* 9, 2 (2017), 91–102.

[36] Srinivasan Ramesh, Swann Perarnau, Sridutt Bhalachandra, Allen D. Malony, and Peter H. Beckman. 2019. Understanding the Impact of Dynamic Power Capping on Application Progress. In *2019 IEEE International Parallel and Distributed Processing Symposium, IPDPS 2019, Rio de Janeiro, Brazil, May 20-24, 2019.* IEEE, 793–804.

[37] M Aaron Skinner and Eve C Ostriker. 2010. The Athena astrophysical magnetohydrodynamics code in cylindrical geometry. *The Astrophysical Journal Supplement Series* 188, 1 (2010), 290.

[38] Peter Thoman, Philip Salzmann, Biagio Cosenza, and Thomas Fahringer. 2019. Celerity: High-level c++ for accelerator clusters. In *Euro-Par 2019: Parallel Processing: 25th International Conference on Parallel and Distributed Computing, Göttingen, Germany, August 26–30, 2019, Proceedings 25.* Springer, 291–303.

[39] Ananta Tiwari, Michael Laurenzano, Joshua Peraza, Laura Carrington, and Allan Snavely. 2012. Green Queue: Customized Large-Scale Clock Frequency Scaling. In *2012 Second International Conference on Cloud and Green Computing.* 260–267. https://doi.org/10.1109/CGC.2012.62

[40] Bart van der Holst, Rony Keppens, and Zakaria Meliani. 2008. A multidimensional grid-adaptive relativistic magnetofluid code. *Computer Physics Communications* 179, 9 (2008), 617–627.

[41] Giulio Vistoli, Candida Manelfi, Carmine Talarico, Anna Fava, Arieh Warshel, Igor V. Tetko, Rossen Apostolov, Yang Ye, Chiara Latini, Federico Ficarelli, Gianluca Palermo, Davide Gadioli, Emanuele Vitali, Gaetano Varriale, Vincenzo Pisapia, Marco Scaturro, Silvano Coletti, Daniele Gregori, Daniel Gruffat, Edgardo Leija, Sam Hessenauer, Alberto Delbianco, Marcello Allegretti, and Andrea R. Beccari. 2023. MEDIATE - Molecular DockIng at homE: Turning collaborative simulations into therapeutic solutions. *Expert Opinion on Drug Discovery* 18, 8 (2023), 821–833. https://doi.org/10.1080/17460441.2023.2221025 arXiv:https://doi.org/10.1080/17460441.2023.2221025 PMID: 37424369.

[42] Emanuele Vitali, Federico Ficarelli, Mauro Bisson, Davide Gadioli, Massimiliano Fatica, Andrea R. Beccari, and Gianluca Palermo. 2022. GPU-optimized Approaches to Molecular Docking-based Virtual Screening in Drug Discovery: A Comparative Analysis. arXiv:2209.05069 [cs.DC]

[43] Gene Wu, Joseph L. Greathouse, Alexander Lyashevsky, Nuwan Jayasena, and Derek Chiou. 2015. GPGPU performance and power estimation using machine learning. In *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA).* 564–576. https://doi.org/10.1109/HPCA.2015.7056063

[44] Elizabeth Yuriev, Jessica Holien, and Paul A. Ramsland. 2015. Improvements, trends, and new ideas in molecular docking: 2012-2013 in review. *Journal of Molecular Recognition* 28, 10 (2015), 581–604.

[45] Huazhe Zhang and Henry Hoffmann. 2018. Performance & Energy Tradeoffs for Dependent Distributed Applications Under System-Wide Power Caps. In *Proceedings of the 47th International Conference on Parallel Processing* (Eugene, OR, USA) *(ICPP '18).* Association for Computing Machinery, New York, NY, USA, Article 67, 11 pages. https://doi.org/10.1145/3225058.3225098

[46] Huazhe Zhang and Henry Hoffmann. 2019. PoDD: Power-Capping Dependent Distributed Applications. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Denver, Colorado) *(SC '19).* Association for Computing Machinery, New York, NY, USA, Article 28, 23 pages. https://doi.org/10.1145/3295500.3356174

[47] Udo Ziegler. 2008. The NIRVANA code: Parallel computational MHD with adaptive mesh refinement. *Computer Physics Communications* 179, 4 (2008), 227–244.