EVALUATING EFFECTS OF APPLICATION BASED AND AUTOMATIC ENERGY SAVING STRATEGIES ON NWCHEM

Vaibhav Sundriyal
Department of Modeling,
Simulation, and Visualization Engineering
Old Dominion University
Norfolk, VA, 23529
vsundriy@odu.edu

Ellie Fought
Department of Chemistry
Iowa State University
Ames, IA 50010, USA
foughtel@iastate.edu

Masha Sosonkina
Department of Modeling,
Simulation, and Visualization Engineering
Old Dominion University
Norfolk, VA, 23529
msosonki@odu.edu

Theresa L. Windus
Department of Chemistry
Iowa State University
Ames, IA 50010, USA
twindus@iastate.edu

ABSTRACT

High-performance application developers are becoming increasingly aware of effects of the increasing energy consumption on the costs and reliability of modern computing systems. A traditional way to achieve energy savings is by changing the processor frequency dynamically during application execution. Several techniques have been proposed in the past at application, library, and transparent level. In this work, the effect of two such techniques, at application and transparent levels, are evaluated in terms of their effects on the execution time and energy consumption for different algorithms in the quantum chemistry package NWChem. Experimental results depict that there is no clear winner between the two methods since the transparent-level makes decisions without intimate knowledge of the application while the strategy based solely on application does not take into the account the platform characteristics at the runtime. Hence, it is argued that the best strategy would be a hybrid of the two levels.

Keywords: DVFS, Energy, Power, NWChem, Oversubscription.

1 INTRODUCTION

Power consumption has become a major concern in the design of modern computing systems due to the fact that power consumption varies as the product of the square of the voltage and the operating frequency which are interdependent as well. For the current topmost petascale computing platforms in the world, it is typical to consume power on the order of several megawatts, which at current prices may cost on the order of several million dollars annually. To address this challenge, power and energy optimizations are needed in modern computing platforms at all levels: application, system software, and hardware.

SpringSim-HPC 2017, April 23-26, Virginia Beach, VA, USA

The CPU and memory subsystems are the major energy consumers in a computing system, together contributing about 50-70% (Ge, Feng, Song, Chang, Li, and Cameron 2010) of the node power. To tackle the issue of increasing power consumption during application execution, DVFS (Dynamic Voltage and Frequency Scaling) has been one of the most used techniques in recent times. The current generation of Intel processors provides different power states (P-states) DVFS. More specifically, the Intel "Haswell-EP" microarchitecture provides a total of ten P-states. The delay of switching from one state to another depends on the relative ordering of the current and desired states, as discussed, e.g., in (Park, Shin, Chang, and Pedram 2010). The user may write a value to model-specific registers (MSRs) to change the P-state of the processor. The "Haswell" micro-architecture estimates power and energy consumption of the CPU and memory through the built-in MSRs, which certainly facilitates power-measurement efforts. The terms PF and MF are used to denote processor-frequency scaling and memory-frequency scaling, respectively, in this work.

In general, energy savings can be obtained by making judicious use of DVFS in modern computing systems at application/library/transparent levels. Since modifying the application itself can be tedious and does not provide enough information about the application characteristics on a particular hardware platform, this approach in general has been ignored by the researchers in the past.

In this work, an application and a transparent level DVFS based technique is evaluated in terms of its energy saving potential on NWChem on two different hardware platforms providing different DVFS applicability domains. The obtained results provide deep insight into the behavior of the different MP2 algorithms when operated under a specific DVFS based strategy along with the differing hardware platforms. The rest of the paper is organized as follows. Section 2 describes the related work and Section 3 provides an overview of the NWChem package. Section 4 discusses the salient features of the two DVFS based techniques. Section 5 details the experimental hardware and the results, while Section 6 concludes the paper and discusses the future work.

2 RELATED WORK

With regards to the library/automatic level application of DVFS, mainly two approaches exist. The first approach focusses on identifying stalls during the execution by measuring architectural parameters like instructions retired and memory accesses per second from performance counters as proposed in (Ge, Feng, Feng, and Cameron 2007, Hsu and Feng 2005, Huang and Feng 2009). The adaptive frequency scaling approach "ondemand" present in the linux kernel was compared to automatic strategy in (Huang and Feng 2009) and it was shown to be considerably less effective. The other approaches primarily focus on scaling processor frequency during slack or communication operations during application runtime. The techniques in the past have targeted communication intervals in parallel applications that use either explicit message passing (Freeh and Lowenthal 2005, Lim, Freeh, and Lowenthal 2006) or global address-space primitives (Vishnu, Song, Marquez, Barker, Kerbyson, Cameron, and Balaji 2010) and then scales the frequency for those intervals. Oversubscribing the processor cores (Iancu, Hofmeyr, Blagojevic, and Zheng 2010) is another technique which can be used to reduce execution time and lower power consumption of a parallel application. Since the efficacy of oversubscription is highly dependent on the inherent nature of the application and operating system features, it has not been widely used as a dedicated technique to improve energy efficiency in modern computing systems. An application based DVFS strategy focusing on NWChem was proposed in (Sundriyal, Fought, Sosonkina, and Windus 2016) which applied either PF of MF depending on the phase of the execution.

David *et al.* (David, Fallin, Gorbatov, Hanebutte, and Mutlu 2011) propose a memory-frequency scaling mechanism based on a memory bandwidth usage estimate such that frequency scaling is applied whenever memory-bandwidth usage goes below a certain level. In (Ge, Feng, He, and Zou 2016) authors study the impact of power allocation to different components of a computing system and attempt to determine the

optimal allocation budget using derived heuristics and profiling. The Intel Haswell-EP processor generation is studied in terms of its new features in micro-architectural innovations and frequency scaling especially the per core frequency scaling capability, in (Hackenberg, Schöne, Ilsche, Molka, Schuchart, and Geyer 2015). An algorithm to automatically determine performance and power models of parallel applications without relying on previous execution data was proposed in (Sensi, Torquati, and Danelutto 2016).

3 NWCHEM OVERVIEW

NWChem (Valiev, Bylaska, Govind, Kowalski, Straatsma, Dam, Wang, Nieplocha, Apra, Windus, and de Jong 2010) is an ab initio computational chemistry software package that provides many methods for computing the properties of molecular and periodic systems using standard quantum mechanical descriptions of the electronic wavefunction or density. It is a scalable, portable, open-source computational chemistry software package, which was designed to be used on multiple platforms and with different computer hardware. However, the prime design target is high performance computers and efficient use of all of the hardware components on those platforms.

Møller-Plesset Second Order Perturbation Theory (Knowles, Andrews, Amos, Handy, and Pople 1991, Lauderdale, Stanton, Gauss, Watts, and Bartlett 1991), or MP2, is an electron correlation method used in association with the self-consistent field, or Hartree Fock (HF) method, in quantum chemistry. MP2 is based on the foundational Rayleigh-Schrödinger perturbation theory using the Fock operator as the unperturbed operator, and is one of the most widely used quantum mechanical correlation methods available. The HF method is an iterative procedure that assumes that each electron is in a mean field of the other electrons and, therefore, ignores the instantaneous correlations of the electrons with one another. The MP2 method is a correction to include this correlation energy. There are three common MP2 algorithms used in NWChem: semi-direct, direct, and resolution of the identity MP2 (Feyereisen, Fitzgerald, and Komornicki 1993, Bernholdt and Harrison 1996), or RI-MP2.

In two of the MP2 methods (direct and semi-direct) within NWChem, an order N^5 transformation of the atomic orbital integrals to the molecular orbital integrals is required, where N is the number of atomic orbital basis functions for the molecule of interest. This is the computational bottleneck for these MP2 energy methods. The direct MP2 performs all calculations and stores all of its integrals and other data to local memory. Any integrals that cannot be stored in memory are recalculated when they are subsequently needed in the calculations. For semi-direct MP2 some transformed integrals get stored in memory or recalculated and some (expensive ones) get written to disk. The semi-direct method is widely used, particularly because of its lower memory requirements. Both direct MP2 and semi-direct MP2 use very similar mathematical algorithms in their calculations and mostly differ in the integral storage.

RI-MP2, on the other hand, uses the resolution of the identity mathematical approximation to transform four-center integrals to three-center integrals. Each of the three-center integrals requires less time to compute, but results in more terms to calculate. RI is only exactly true when the basis set is complete which in general, is not possible for a finite basis set and so a large auxiliary basis sets is used to have a more complete basis set for the RI part. This approximation also changes the order N^5 MO integral transformation into an order N^4 operation. However, the final energy evaluations is still order N^5 due to the extra multiplications involved to approximate the four-center integrals. The NWChem implementation also uses on order N^2 total memory and order N^3 disk storage (much less than the semi-direct method, but more than the direct method).

In this work, the test cases are referred to by the system size, i.e., the number of atoms in the system, and by the MP2 algorithm employed: The test-name suffix “s” stands for semi-direct, “d” for direct, and “r” for RI-MP2. The system with 40 atoms having 335 standard and 800 auxiliary basis functions per molecule was chosen for the experimental evaluation. Although a 55-atom system was used in the authors’ work (Sundriyal, Fought, Sosonkina, and Windus 2016) to experiment with large calculations, the 40-atom input

was chosen here instead, to provide fair comparisons of the platforms used in terms of efficient executions of the calculations. Conventional SCF was used in all the experiments conducted. Each test was executed at least three times to determine reproducibility of the results and the average values of those runs are presented in the paper.

4 OVERVIEW OF THE STRATEGIES

4.1 Application Based

A "hands-on" instrumentation (denoted as "HI") of the code DVFS based strategy was proposed in (Sundriyal, Fought, Sosonkina, and Windus 2016) with steps as follows to save energy:

- `If` in the startup section, `then` lower the MF while keeping PF at the maximum.
- `If` in the HF section, `then` lower the PF while keeping MF at the maximum.
- `If` in the MP2 section, `then` lower the MF and set the PF to the maximum.

The proposed HI strategy was further refined based on the range of the available PFs considered. The *aggressive* HI uses the minimum values of the respective, processor and memory frequencies to aggressively target possible energy savings. On the other hand, the *moderate* HI strategy uses the middle value in the range of available processor and memory frequencies. Since the results in (Sundriyal, Fought, Sosonkina, and Windus 2016) depicted that the *aggressive* HI provided higher energy savings compared to *moderate* HI, the *aggressive* HI is considered for evaluation purposes in this work.

4.2 Automatic Strategy

An automatic runtime strategy proposed in (Sundriyal and Sosonkina 2016b) is considered here as transparent to the application. This strategy predicts the micro-operations retired at different processor and memory frequencies along with the system power consumption and selects the appropriate processor–memory frequency pairs that minimize total energy consumption while satisfying the performance-loss constraint. The runtime strategy makes use of timeslices and uses history-window approach to predict application future behavior.

The base performance model that has been employed in the automatic strategy is

$$\mu\tau(i,j) = \frac{f_p(i)}{\mathrm{CPM_{exe}} + \mathrm{MLIF}(j) \times \alpha \times \mathrm{MAPM} \times \beta \times \frac{f_p(i)}{f_p(1)}} \,. \tag{1}$$

where

- $\mu\tau(i,j)$ is the actual number of micro-operations retired per second at processor frequency $f_p(i)$ and memory frequency $f_m(j)$.
- $\mathrm{CPM_{exe}}$ is the number of cycles per micro-operations retired barring the memory accesses in a second.
- α ($0 \leq \alpha \leq 1$) is the OOO (out-of-order) overlap factor, which determines the extent of memory stalls overlapped with execution cycles.
- MAPM is the number of memory accesses per micro operation retired in a second.
- β is the number of cycles corresponding to the memory-access latency.

- MLIF(j) is the memory latency increase factor at the memory frequency $f_m(j)$ which depicts the relative increase in memory latency at memory frequency $f_m(j)$ compared to the highest memory frequency $f_m(1)$.

Using this performance model and an adaptive mechanism to adjust dynamically the memory and compute-intensity of an application, the strategy was shown delivering significant energy savings for the SPEC CPU^{TM} 2006 and NAS parallel benchmarks.

5 EXPERIMENT SETUP AND RESULTS

Two different testbeds are used in this work, each of them having a particular DVFS granularity and subsequent effective frequency. DVFS granularity refers to the grouping of cores in a processor socket with respect to independent frequency and voltage scaling domains. In (Sundriyal and Sosonkina 2016a), it was determined that an energy saving strategy will be ineffective on an application which has variable workload nature on different cores, executing on a hardware platform which doesn't have per core DVFS granularity.

The effective frequency f_{eff} can be defined as the frequency experienced by a multicore node, when each core i ($i = 0, \ldots, n-1$) is in a certain P-state f_i. For expressing the effective frequencies of the two platforms used in this work, n core testbeds are considered, where n is even, and supporting a specific level of the DVFS application.

1. `Marquez`[1] is comprised of a single node with two Intel Xeon CPU E5-2630 v3 "Haswell-EP" eight-core processors (two sockets) with 32 GB of main memory. The Intel Xeon E5-2630 v3 processor provides thirteen P-states ranging from 1.2 to 2.0 GHz. The Intel Xeon CPU E5-2630 v3 ("Haswell-EP") processor on Marquez has multiple fully integrated voltage regulators providing an individual voltage for each core which results in per core P-states (PCPS) (Hackenberg, Schöne, Ilsche, Molka, Schuchart, and Geyer 2015). This enables independent frequency scaling "per core" rather than "per-socket". (Marquez is funded and operated by Old Dominion University.)
2. `Styx` has an Intel i5-4590 "Haswell" quad-core processor and 8 GB of main memory with timing specification 9-9-9-24. The processor frequency ranged from 3.3 GHz to 0.8 GHz; for memory, the frequency range was from 1.6 GHz to 0.8 GHz. The processor frequency was modified by writing a specific alpha-numeric value to model specific register (MSR) IA32_PERF_CTL.

The *effective frequency* f_{eff} for the two platforms can be expressed as,

1. For the PCPS level (as in Marquez),
$$f_{\text{eff}} = f_i \,. \tag{2}$$
2. For the socket level (as in Styx),
$$f_{\text{eff}} = \max(f_0, f_1, \ldots, f_{n-1}) \,. \tag{3}$$

The experiments were conducted on a single node each of Styx and Marquez. For measuring the node power consumption, a Wattsup power meter was used with a sampling rate of 1 Hz. The processor frequency was modified by writing a specific alpha-numeric value to model specific register (MSR) IA32_PERF_CTL. A performance loss of 10% was chosen for the *automatic* strategy. Table 1 depicts the value of the relevant parameters needed for Equation (1) for the two hardware platforms. An important point to notice here is the significantly lower value of memory latency and out-of-order overlap factor in case of Marquez compared to Styx which can potentially turn a memory intensive task on Styx to a compute intensive task on Marquez.

Table 1: Parameters for Marquez and Styx relevant for the automatic strategy.

	Marquez	Styx
OOO Overlap Factor-α	0.28	0.58
Memory Latency (in Cycles)-β	200	270
Processor Frequency Range	1.2-2.4 GHz	0.8-3.3 GHz
Memory Frequency Range	1.33-2.666 GHz	0.8-1.6 GHz

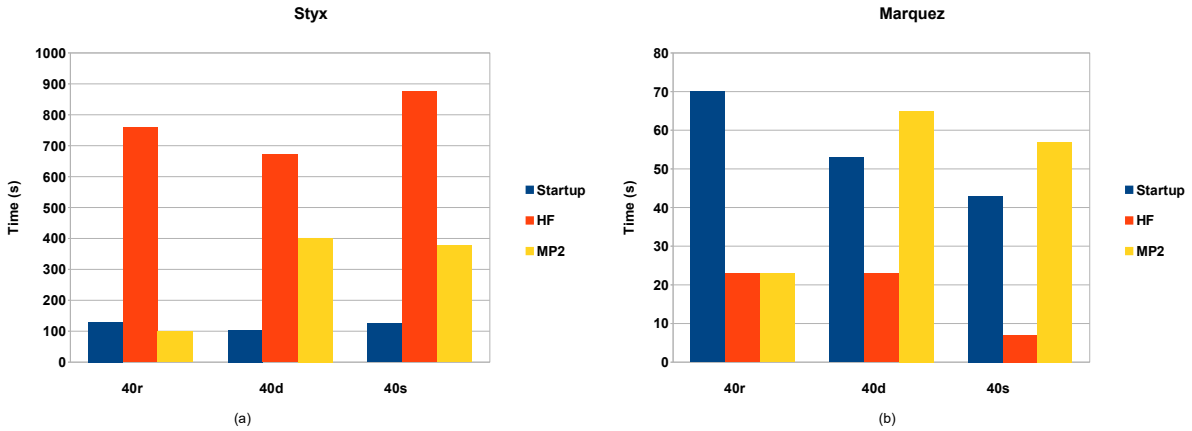

Figure 1: Execution time for different sections in the 40-atom system solved by three MP2 algorithms on (a) Styx and (b) Marquez, where 40r, 40d and 40s refer to the RIMP2, Direct and Semi-direct algorithms, respectively.

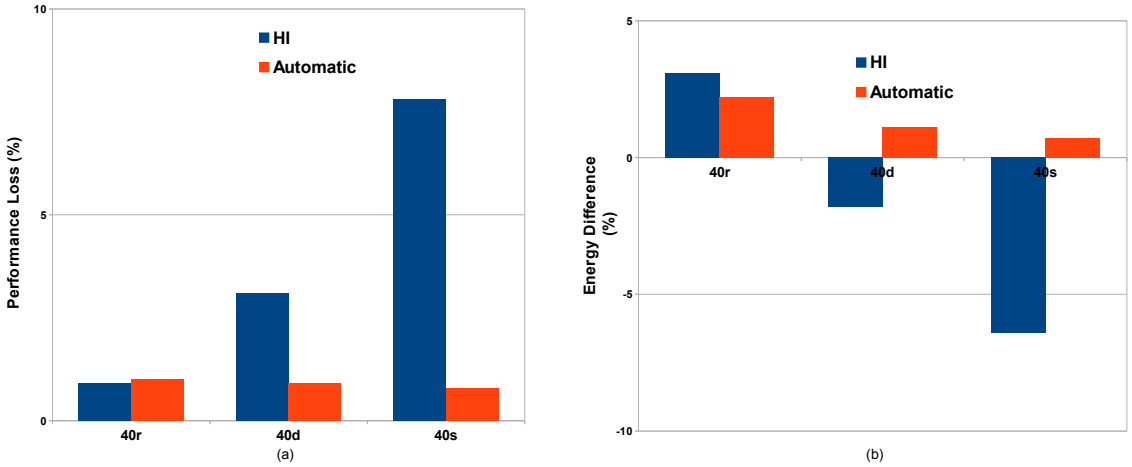

Figure 2: For Styx, total performance degradation (a) and energy difference (b) for the 40-atom system solved by the three MP2 algorithms under the HI and Automatic strategies.

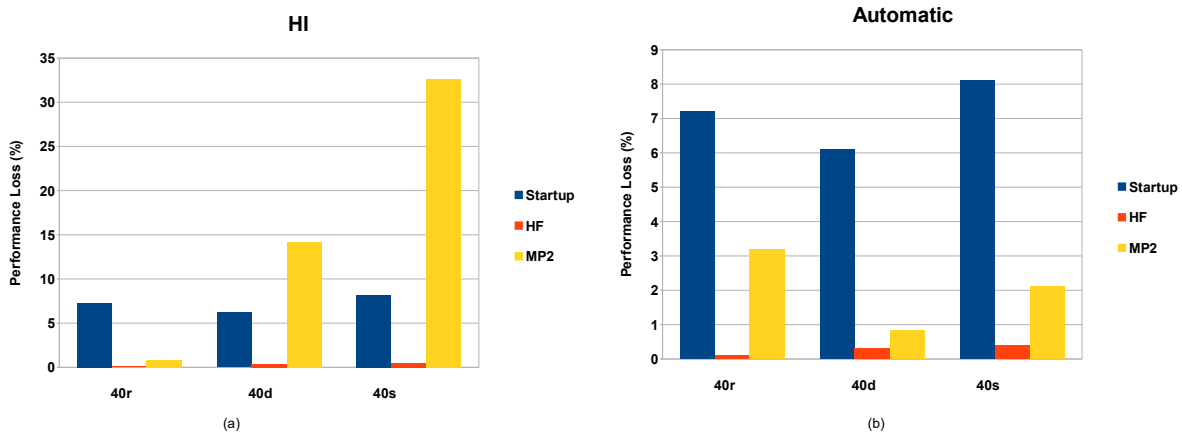

Figure 3: For Styx, breakdown of performance degradation by section for the 40-atom system solved by three MP2 algorithms operating under (a) HI and (b) Automatic strategy.

5.1 Section Timings

Figure 1 depicts the execution time of different sections for the 40-atom system for the three algorithms executing on Styx (Fig. 1(a)) and Marquez (Fig. 1(b)). It can be observed that the startup section tends to take nearly the same time for all the three algorithms on Styx whereas on Marquez, it is variable. As for the HF section, it has the highest proportion of the execution time for all the inputs on Styx. On Marquez, the startup section dominates the execution time for the RIMP2 inputs, but in the case of direct and semi-direct, the HF section executes for slightly lower time than MP2 section. Considering the MP2 section, RIMP2 is the fastest of the three followed by semi-direct and direct.

5.2 Styx

Figure 2 shows the performance loss and change in energy consumption for the three MP2 algorithms when operated under the *HI* and *Automatic* strategies compared to the baseline case in which both PF and MF were set at their highest levels on *Styx*. The section wise performance loss for the 40-atom system solved by the three MP2 algorithms operating under the *HI* and *Automatic* strategies is provided in Fig. 3. It can be noted here that a negative value of energy difference denotes an overall increase in energy consumption.

In the case of RIMP2 algorithm, the *HI* and *Automatic* strategy both result in a moderate performance loss of $\sim$1%. The startup section in RIMP2 shows fairly compute intensive behavior enabling both *HI* and *Automatic* strategies to scale the memory frequency to its lowest value (0.8 GHz). During the HF section in RIMP2 algorithm, the *Automatic* strategy mostly executes at 2.8-3.0 GHz PF since the nature of the workload in terms of memory accesses per instruction remains quite variable with most of the time processor remaining in an idle state. During the MP2 stage in RIMP2, the *Automatic* strategy chooses MF of 1.066 GHz, compared to 0.8 GHz in the *HI* strategy.

For all the three algorithms, the HF section tends to be mostly I/O intensive with minor compute intensive phases in between. Therefore, reducing the PF during the HF section contributes to negligible overall performance degradation for the three algorithms operating under the *HI* strategy as noticed in Fig. 3(a). Since the behavior of HF section remains the same across the three algorithms in terms of workload, the *Automatic* strategy, ends up executing the HF mostly at the highest PF and MF and does not result in any

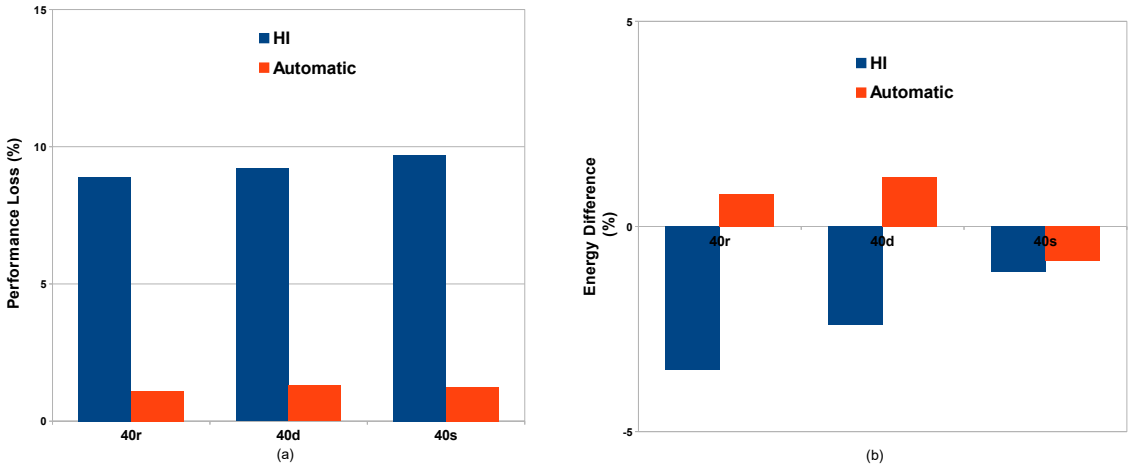

Figure 4: For Marquez: Total performance degradation (a) and energy difference (b) for the 40-atom system solved by the three MP2 algorithms under the HI and Automatic strategies.

significant performance degradation. The MP2 section in RIMP2 tends to be much more compute intensive compared to direct and semi-direct MP2 and hence reducing the MF to its lowest value under *HI* results in negligible performance degradation. On the other hand, majority of the performance degradation for the overall direct and semi-direct execution is due to significant performance loss in the respective MP2 sections under the *HI* strategy. To summarize, average performance degradations across the three algorithms are 4% and 0.88% for the *HI* and *Automatic* strategies, respectively.

With regards to energy savings, the *HI* and *Automatic* strategies reduce the energy consumption by 3.1% and 2.2%, respectively for the RIMP2 algorithm. However, the direct(-1.8%) and semi-direct (-6.4%) algorithms experience an overall increase in energy consumption when operated under the *HI* strategy due to the fact that they don't offer neither substantially memory or CPU intensive behavior to benefit from either PF or MF, respectively. Also, under the *HI* strategy, the MF scaling applied to the MP2 section in direct and semi-direct algorithms degrades overall performance significantly. The *Automatic* strategy remains somewhat conservative with respect to frequency scaling throughout the direct and semi-direct execution and only applies MF of 1.33 GHz during the MP2 section. The average energy savings across the three algorithms are -1.8% and 1.33% for the *HI* and *Automatic* strategies, respectively.

The 40 atoms inputs behave largely similar to the 55 atoms system used in (Sundriyal, Fought, Sosonkina, and Windus 2016) in terms of relative timings of HF, SCF and MP2 sections and the effect of PF and MF on these timings as HF and MP2 sections tend to be compute intensive whereas SCF is largely memory/IO intensive. The difference mainly exists in the extent to which different sections in the three algorithms are affected by PF and MF. For example, the inability of *HI* strategy to provide energy savings for 40d and 40s inputs mainly comes from the resulting higher performance degradation through application of MF in the MP2 section, compared to the 55 atom inputs. At this point, the change in processor and memory intensity of these algorithms with varying system size is under investigation.

5.3 Marquez

Figure 4 depicts the performance loss and change in energy consumption for the three MP2 algorithms when operated under the *HI* and *Automatic* strategies compared to the baseline case in which both PF and MF were set at their highest levels on Marquez. The performance loss broken down for different sections of the 40-atom system execution solved by the three MP2 algorithms operating under the *HI* and *Automatic* strategies

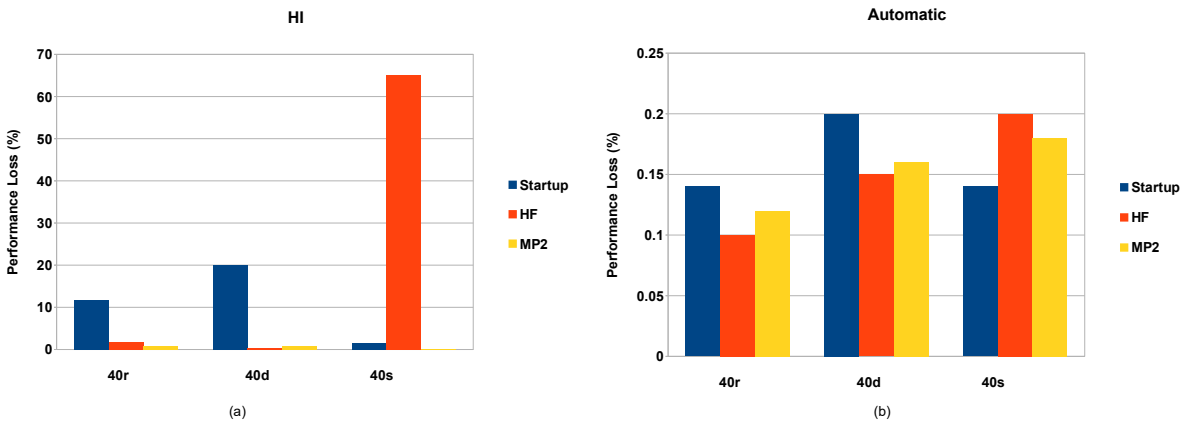

Figure 5: For Marquez, breakdown of performance degradation by section for the 40-atom system solved by three MP2 algorithms operating under (a) HI and (b) Automatic strategy.

on Marquez is shown in Fig. 5. Due to a lower OOO overlap factor and higher speed and bandwidth provided by the DDR4 compared to the DDR3 on Styx, the three inputs tend to be relatively CPU intensive compared to their execution behavior on Styx. Consequently, the *HI* strategy results in considerable performance loss for all the three algorithms averaging 8.5%.

The *Automatic* strategy on the other hand, being aware of the change in platform and the relevant hardware parameters, executes the three inputs at the highest PF most of the time with execution dipping to 1.2 GHz PF in short bursts for direct and RIMP2 algorithms whereas the semi-direct algorithm tends to be the most compute intensive and no PF is applied to it. Also, no MF is applied to the three algorithms since the corresponding reduction in DRAM power is not enough to compensate for the performance loss incurred as per *Automatic* strategy. It can be seen in Fig. 5 that section-wise performance degradation is significantly lower for *Automatic* compared to *HI* and it mostly comes from profiling on Marquez. Therefore, the incurred performance loss for the *Automatic* strategy is much less compared to the *HI* for the three algorithms. More specifically, the average performance loss for the three algorithms is 1.2% when operated under the *Automatic* strategy.

Due to aggressive application of both PF and MF and the subsequent reduction in power which is insufficient to compensate for the performance loss, the *HI* strategy increases the energy consumption for all three algorithms, averaging -2.3%. The *Automatic* strategy on the other hand is extremely conservative in applying both PF and MF and saves energy for both RIMP2 (0.8%) and direct(1.2%). No frequency scaling is applied to the semi-direct algorithm under the *Automatic* strategy so there is a slight increase in energy consumption due to the minor profiling overhead.

6 CONCLUSIONS AND FUTURE WORK

This work studied the energy saving potential of the two variants of the DVFS based strategies evaluating three different algorithms in NWChem: RI-MP2, direct MP2, and semi-direct MP2, where the last two methods differ in the treatment of memory and disk storage of the integrals and intermediate data. The two DVFS based strategies were chosen such that one of them operates transparently to the application (automatic) and the other manages changes in the applications source code itself to insert calls to frequency scaling (HI).

The strategies were evaluated on two different platforms which differ in the processor generation, number of cores, processor frequency and amount and speed of memory. It was observed that the workload behavior of the three algorithms changed on the two platforms due to the difference in the hardware parameters and the *Automatic* strategy adapted to these changes in a better manner compared to *HI*. Overall, the *Automatic* strategy was the more conservative of the two in terms of minimizing performance degradation whereas the *HI* strategy in some cases ended up saving more energy albeit at the cost of application performance. In fact, when calculating the 40-atom system, NWChem doesn't provide enough opportunity to apply frequency scaling on the two platforms. So, even employing an *HI* moderate strategy, would not make much of a difference except it would only decrease the increase in energy savings noticed with the aggressive strategy by a bit. Since NWChem is one of the prominent quantum chemistry applications and enjoys a large user community, the findings of this paper are beneficial to an important scientific domain (ab initio quantum chemistry), which is a focus of a large number of HPC packages, such as GAMESS (Schmidt, Baldridge, Boatz, Elbert, Gordon, Jensen, Koseki, Matsunaga, Nguyen, Su, Windus, Dupuis, and Montgomery 1993) and PSI4 (Turney, Simmonett, Parrish, Hohenstein, Evangelista, Fermann, Mintz, Burns, Wilke, Abrams, Russ, Leininger, Janssen, Seidl, Allen, Schaefer, King, Valeev, Sherrill, and Crawford 2012), each having computational stages as considered in this paper.

Each of the two strategies that have been studied in this work have their own pros and cons. While the *HI* strategy is relatively easy to deploy and apply, it suffers from the fact that it is not aware of the nature of the underlying platform since the extent of application of frequency scaling is predetermined. On the other hand, the *Automatic* strategy constantly profiles the application to collect runtime information used in the decision making for frequency scaling but it can be difficult to deploy and can also suffer when the underlying hardware platform is changed since it relies on hardware performance counters. Future work would focus on getting the best of both worlds by combining these two strategies into a hybrid strategy. The hybrid strategy would consist of relevant hardware parameter information of a base platform. Depending on the platform on which NWChem would be executing, the hybrid strategy would adjust the frequency levels for the three sections by comparing the relevant hardware parameters of the base platform to the current platform. In this manner, appropriate frequencies would be chosen for the startup, HF and MP2 sections of the code without actually transparently profiling the application.

REFERENCES

Bernholdt, D. E., and R. J. Harrison. 1996. "Large-scale Correlated Electronic Structure Calculations: The RI-MP2 Method on Parallel Computers". *Chem. Phys. Lett.* vol. 250, pp. 477–484.

David, H., C. Fallin, E. Gorbatov, U. Hanebutte, and O. Mutlu. 2011. "Memory Power Management via Dynamic Voltage/Frequency Scaling". In *Proceedings of the 8th ACM International Conference on Autonomic Computing*, pp. 31–40.

Feyereisen, M., G. Fitzgerald, and A. Komornicki. 1993. "Use of Approximate Integrals in ab initio Theory. An Application in MP2 Energy Calculations". *Chem. Phys. Lett.* vol. 208, pp. 359.

Freeh, V., and D. Lowenthal. 2005. "Using Multiple Energy Gears in MPI Programs on a Power-Scalable Cluster". In *Proceedings of the Tenth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pp. 164–173.

Ge, R., X. Feng, W. Feng, and K. Cameron. 2007, Sep.. "CPU MISER: A Performance-Directed, Run-Time System for Power-Aware Clusters". In *Parallel Processing, 2007. ICPP 2007. International Conference on*, pp. 18.

Ge, R., X. Feng, Y. He, and P. Zou. 2016, Aug. "The Case for Cross-Component Power Coordination on Power Bounded Systems". In *2016 45th International Conference on Parallel Processing (ICPP)*, pp. 516–525.

Ge, R., X. Feng, S. Song, H. Chang, D. Li, and K. Cameron. 2010. "PowerPack: Energy Profiling and Analysis of High-Performance Systems and Applications". *Parallel and Distributed Systems, IEEE Transactions on* vol. 21, pp. 658–671.

Hackenberg, D., R. Schöne, T. Ilsche, D. Molka, J. Schuchart, and R. Geyer. 2015, May. "An Energy Efficiency Feature Survey of the Intel Haswell Processor". In *Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International*, pp. 896–904.

Hsu, C., and W. Feng. 2005, nov. "A Power-Aware Run-Time System for High-Performance Computing". In *Supercomputing, 2005. Proceedings of the ACM/IEEE SC 2005 Conference*, pp. 1.

Huang, S., and W. Feng. 2009, May. "Energy-Efficient Cluster Computing via Accurate Workload Characterization". In *Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium on*, pp. 68–75.

Iancu, C., S. Hofmeyr, F. Blagojevic, and Y. Zheng. 2010. "Oversubscription on Multicore Processors". In *Parallel Distributed Processing (IPDPS), 2010 IEEE International Symposium on*, pp. 1–11.

Knowles, P., J. Andrews, R. Amos, N. Handy, and J. Pople. 1991. "Restricted Møller—Plesset Theory for Open-Shell Molecules". *Chem. Phys. Lett.* vol. 186, pp. 130–136.

Lauderdale, W., J. Stanton, J. Gauss, J. Watts, and R. Bartlett. 1991. "Many-body Perturbation Theory with a Restricted Open-shell Hartree—Fock Reference". *Chem. Phys. Lett.* vol. 187, pp. 21–28.

Lim, M., V. Freeh, and D. Lowenthal. 2006. "Adaptive, transparent frequency and voltage scaling of communication phases in MPI programs". In *Proceedings of the 2006 ACM/IEEE conference on Supercomputing*.

Park, J., D. Shin, N. Chang, and M. Pedram. 2010. "Accurate Modeling and Calculation of Delay and Energy Overheads of Dynamic Voltage Scaling in Modern High-Performance Microprocessors". In *2010 International Symposium on Low-Power Electronics and Design (ISLPED)*, pp. 419–424.

Schmidt, M. W., K. Baldridge, J. Boatz, S. Elbert, M. Gordon, J. Jensen, S. Koseki, N. Matsunaga, K. Nguyen, S. Su, T. Windus, M. Dupuis, and J. J. Montgomery. 1993, November. "General atomic and molecular electronic structure system". *J. Comput. Chem.* vol. 14, pp. 1347–1363.

Sensi, D. D., M. Torquati, and M. Danelutto. 2016, December. "A Reconfiguration Algorithm for Power-Aware Parallel Applications". *ACM Trans. Archit. Code Optim.* vol. 13 (4), pp. 43:1–43:25.

Sundriyal, V., E. Fought, M. Sosonkina, and T. L. Windus. 2016. "Power Profiling and Evaluating the Effect of Frequency Scaling on NWChem". In *Proceedings of the 24th High Performance Computing Symposium*, HPC '16, pp. 19:1–19:8. San Diego, CA, USA, Society for Computer Simulation International.

Sundriyal, V., and M. Sosonkina. 2016a. "Effect of Frequency Scaling Granularity on Energy-Saving Strategies". *Submitted to the International Journal of High Performance Computing Applications.*.

Sundriyal, V., and M. Sosonkina. 2016b. "Joint Frequency Scaling of Processor and DRAM". *The Journal of Supercomputing* vol. 72 (4), pp. 1549–1569.

Turney, J., A. Simmonett, R. Parrish, E. Hohenstein, F. Evangelista, J. Fermann, B. Mintz, L. Burns, J. Wilke, M. Abrams, N. Russ, M. Leininger, C. Janssen, E. Seidl, W. Allen, H. Schaefer, R. King, E. Valeev, C. Sherrill, and T. Crawford. 2012. "Psi4: An Open-source ab initio Electronic Structure Program". *Wiley Interdisciplinary Reviews: Computational Molecular Science* vol. 2 (4), pp. 556–565.

Valiev, M., E. Bylaska, N. Govind, K. Kowalski, T. Straatsma, H. V. Dam, D. Wang, J. Nieplocha, E. Apra, T. Windus, and W. de Jong. 2010. "NWChem: A Comprehensive and Scalable Open-source Solution for Large Scale Molecular Simulations". *Computer Physics Communications* vol. 181 (9), pp. 1477 – 1489.

Vishnu, A., S. Song, A. Marquez, K. Barker, D. Kerbyson, K. Cameron, and P. Balaji. 2010. "Designing Energy Efficient Communication Runtime Systems for Data Centric Programming Models". In *Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*, GREENCOM-CPSCOM '10, pp. 229–236. Washington, DC, USA, IEEE Computer Society.

AUTHOR BIOGRAPHIES

VAIBHAV SUNDRIYAL is working as a Research Scientist at ODU Research Foundation. He holds a PhD in Computer Engineering from Iowa State University. His research interests lie in computer architecture and power management especially in HPC systems. His email address is vsundriy@odu.edu.

ELLIE FOUGHT is a graduate student in Department of Chemistry at Iowa State University working towards her Ph.D. She is working on incorporating power considerations when optimizing quantum mechanical algorithms. She has also examined NMR properties for organic complexes. Her email address is foughtel@iastate.edu.

MASHA SOSONKINA received her Ph.D. from Virginia Tech, and currently is Professor in the Department of Modeling, Simulation and Visualization Engineering at the Old Dominion University. Her research interests include applied computational mathematics, high performance computing and chemical and biological sciences. Her email address is msosonki@odu.edu.

THERESA L. WINDUS is a Professor in the Department of Chemistry at Iowa State University. She earned her Ph.D. from Iowa State University in 1993 and did post-doctoral research at Northwestern University. Her research interests include high performance computing and methods for accurate energies. Her email address is twindus@iastate.edu.