Master in High Performance Computing

Coupling of i-PI and ONETEP codes to enable large/petaScale simulations for first principles modeling of both electrons and nuclei for thousands of atoms.

Supervisor(s):
Ali Hassanali, Ph.D
Emiliano Poli,Ph.D
Ivan Girotto

Candidate:
Elliot Sarpong Menkah

3rd edition
2016–2017

Contents

List of Figures

Abstract

Nuclear quantum effects play important roles in systems containing hydrogen. Process that involve the effective elucidation of NQEs occur at finite temperature and can be tackled by Feynman's Path-Integral approach. This involves the quantum mechanical treatment of both electrons and nuclei. Advanced techniques in molecular dynamics coupled with high performance computing resources can aid such studies. The goal of this project is to make available the platform for the studies of extended systems in which quantifying quantum fluctuations is of optimal importance. We have implemented a server-client model which allows large-scale atomistic simulation by interfacing two programs, ONETEP and i-PI. The implementation will allow the study of processes involving light nuclei. The coupling of these two codes enables petascale simulations for first principle modeling of both electrons and nuclei of thousands of atoms. The application has been successfully enabled on the Intel Knight-Landing partition of the Tier-0 CINECA computing resource and successful tests have been performed. Some preliminary results as well as scalability profiles are shown in this report.

Chapter 1

Introduction

Molecular dynamics simulation has evolved and is continuing its development into a mature technique that can be used to understand a range of atomistic structure-to-function relationship and also help to predict properties and behavior of new materials and compounds[1]. The present advances of molecular dynamics coupled with state-of-art computing power has brought a paradigm shift in compute-intensive scientific investigations: from studying simple structures to extended systems with accurate treatment of the interaction between atoms [1, 2, 3].

In classical molecular dynamics simulations, the nuclei are evolved according to Newton's equations of motions, $F=ma$. Simulations involving *ab initio* molecular dynamics imply the quantum mechanical treatment of the electrons [4, 5, 6] and typically a classical approximation treatment of the nuclei. Quantification of the quantum fluctuations of nuclei has been shown to be important in the behaviour of systems containing hydrogen or light nuclei such as water[7]. Many organic and inorganic systems contain hydrogen , hence, are likely to to be subject to significant nuclear quantum effects. This is observed in the changes in the properties of water when protons(H) are substituted with deuterium(H) and tritium(T). An example is represented by the case of the melting point of heavy water(D_2O) being about 3.82 K higher than that of water(H_2O) and even more pronounced in tritiated water, (T_2O) [3, 7, 8].

The importance of nuclear quantum effects (NQE) is evident considering that the zero-point energy associated with a typical O-H stretch mode is approximately 200 meV whereas the thermal energy at room temperature, k_BT, is about 25 meV. This difference has significant consequences and highlights the need to take into account the quantum behaviour of nuclei.

The goal of this master thesis is to enable peta-scale simulations of Path Integral *ab initio* Molecular Dynamics where both the electrons and nuclei are treated quantum mechanically. This serves two timely goals:

- first, to be able to take advantage and optimize the use of high performance computing resources available.
- second, to perform scientific investigations considering the quantum mechanical nature of both electrons and nuclei.

This feat is obtainable by coupling recent advances in molecular dynamics techniques with prevailing high performance computing(HPC) power. From this thesis standpoint, such process involves the interfacing of the Linear-Scaling Electronic Total Energy Package, ONETEP[9], with the python wrapper, i-PI[3], which allows Path-integral Molecular dynamics simulations to be performed.

1.1 Objectives

In light of what has been discussed above, the objectives of this thesis is as follows:

1. Interface ONETEP(LS-*ab initio* DFT code, Uni. Of Cambridge, Cambridge) with i-PI(Path-Integral Code, EPFL, Lausanne)
2. Port i-PI-ONETEP onto CINECA's Tier-0 HPC system: MARCONI(A2 partition which is powered by Intel's Knight Landing chipset)
3. Enable path-integral calculation at peta-scale

Chapter 2

Background

As previously discussed in Chapter 1, the goals of this project would be achieved by interfacing two scientific packages:

- i-PI [3][1]
- ONETEP[9]

Each one of these packages have different implementations according to their intended purpose.

i-PI is a python wrapper that allows different types of molecular dynamics simulations. Specifically, it permits the execution of molecular dynamics simulations where the nuclei are treated classically (i.e. standard implementation available in almost all the codes) and in another case, where the nuclei is treated quantum mechanically. The advantage of using i-PI is that it is flexible and can be interfaced with with external programs [10].

While the core of i-PI is mainly focused on molecular dynamics simulations and path integral methods, several techniques have been implemented that exploit the benefits of decoupling force evaluation and nuclear positions evolution. Some of these techniques include *replica exchange, ring polymer contraction[11], geometry optimization[12], isotope Fractionation [13]* and the use of different thermostats in simulations [3, 14, 15]. ONETEP is a linear-scaling DFT *ab-initio* code that allows the simulations of systems comprising thousands of atoms. Its structure and implementation makes it the ideal candidate to be employed for large scale simulation on modern HPC platforms.

Since the focus of my thesis is on expanding the scope of ONETEP to do PIMD simulations, the focus of this chapter will be on the most important aspects of the path integral formalism and linear scaling density functional theory(LS-DFT).

[1]Licensing provisions: MIT/GPLv3

2.1 Linear-Scaling Density Functional Theory

2.1.1 Introduction

Density functional theory (DFT) is presently the most successful (and also the most promising) approach to compute the electronic structure of matter. Its applicability ranges from atoms, molecules and solids to nuclei and quantum and classical fluids. In its original formulation, DFT provides the ground state properties of a system, using as key variable the electron density. DFT [16, 17] has made a unique impact [18] on science, well beyond the traditional realms of quantum-mechanical simulations into disciplines such as microelectronics [19], biochemistry [20] and geology [21].

This broad multidisciplinary appeal has its origin in the ability of DFT to provide a sufficiently accurate description of electron correlation for many purposes (i.e.spin polarized systems, multicomponent systems, free energy at finite temperatures, relativistic electrons, time-dependent phenomena, excited states, bosons, molecular dynamics, etc.) at a favorable computational cost (scaling as the cube of the system size N) compared with correlated wave function methods (which typically exhibit N^5 to N^7 scaling [22, 23, 24]).
The origin of the cubic scaling of DFT is due to its formulation by Kohn and Sham:

$$\hat{H}_s\psi_n(\mathbf{r}) = \left[-\frac{1}{2}\nabla^2 + V_{eff}[n](\mathbf{r})\right]\psi_n(\mathbf{r}) = \epsilon_n\psi_n(\mathbf{r}) \tag{2.1}$$

where $V_{eff}[n](\mathbf{r})$ is the effective potential describing the electron in a mean field manner, n($\mathbf{r}$) is the electron density and $\psi_n(\mathbf{r})$ are the so-called Kohn-Sham orbitals. Equation 2.1 has to be solved self-consistently together with :

$$n(\mathbf{r}) = 2\sum f_n \left|\psi_n(\mathbf{r})\right|^2 \tag{2.2}$$

where the factor 2 accounts for spin degeneracy and f_n are the occupation numbers for each Kohn-Sham orbital. For small systems (i.e. small basis set, low number of atoms) equation 2.1 can be solved via direct matrix diagonalization; a procedure that scales N^3. In cases where the basis set or the dimension of the system are large the Kohn-Sham equation can instead be solved relying on iterative methods [25]. In the latter case the orthonrmality between the different $\psi_n(\mathbf{r})$ orbitals needs to be imposed directly:

$$\int d^3 r \psi_m^*(\mathbf{r}) \psi_n(\mathbf{r}) = \delta_{mn} \tag{2.3}$$

Orthonormality has to be enforced on orbital pairs so that the number of constraints is proportional to N^2. In addition every Kohn-Sham state extends over the entire system thence the evaluation of the integral in Eq.3 requires and additional factor N. In conclusion the overall imposition of orthonormality scales as N^3. As previously stated the N^3 scaling is relatively gentle with respect to other techniques (i.e. Coupled Cluster, MP2, GW) however it still presents a limit to the size of the simulations we are capable of; only systems comprising few hundreds atoms can be routinely treated.

Another important factor to consider when discussing the computational efficiency of DFT calculations is the development of modern High Performance Computing. The growth of accessible computational power is nowadays characterized by an increment in the number of processors available. Therefore, to maximize the efficiency of the available computational power, the algorithm used to solve the DFT problem should offer the possibility of an efficient parallelization. Unfortunately, the heavy use of Fast Fourier Transform (FFTs) in standard plane-wave DFT, which requires a so called all to all communication of information across the processors used in the simulation, makes efficient parallelization of the method far from immediate.These limits pushed the scientific community towards the research more efficient, i.e. better parallelizable and scaling, approaches to DFT calculations.
Particular effort has been put in the development of linear-scaling algorithms. Linear scaling methods exploits the "nearsightedness" of quantum many particle systems [26, 27], in addition to the approximations already used in standard DFT. This principle states that in systems comprised by a large number of interacting particles, the potential felt by each particle is dominated by local interactions, with the definition of "locality" depending on the system of interest. In this regard the expression of "locality" has been estimated for semiconducting and insulating systems on the basis of the Density Matrix Formulation of DFT [28, 29, 30]

$$\rho(\mathbf{r}, \mathbf{r'}) \sim e^{-\gamma|\mathbf{r}-\mathbf{r'}|}; \gamma \propto \sqrt{E_{BG}} \tag{2.4}$$

where γ is positive, $\rho(\mathbf{r}, \mathbf{r'})$ is the density matrix and E_{BG} is the band gap of the system considered. This relation shows that for system with a band gap the elements of the density matrix decay exponentially with the distance between them.

A similar relations holds for metallic systems; however in such systems the elements of the density matrix do not decay exponentially but following a power law [30] In practice the application of the "nearsightedness" principle implies the deliberate imposition of a cutoff distance over which the local density will be blind to the perturbation in the potentials. As this cutoff term is decreased the calculation will become cheaper but less accurate. It is then necessary to control the level of accuracy needed to keep any error due to this approximation to an acceptable level. The next few paragraph will introduce the linear scaling DFT method as implemented in ONETEP [9] one of the programs at the base of this work.

2.1.2 Density Matrix Formulation

LS-DFT codes usually employ the density matrix (DM) as central variable [26, 27]. The DM provides in fact a complete description of the fictitious Kohn-Sham electronic density and allows the use of the nearsightedness property explicitly. The DM can be express as:

$$\rho(\mathbf{r}, \mathbf{r'}) = \sum_n f_n \Psi_n(\mathbf{r}) \Psi_n^*(\mathbf{r'}) \tag{2.5}$$

where f_n are the occupancies of the different states and $\Psi_n(\mathbf{r})$ are the Khon-Sham states of the system. By a formal standpoint the DM is the position representation of the projection operator onto the space of occupied states $\hat{\rho}$. Once our problem is reformulated in a DM format, to find the ground state during a DFT calculation three constraints must be respected. First, the DM must be normalized in order to recover the total number of electrons of the system:

$$N = Tr(\rho) \tag{2.6}$$

where N is the number if electrons and Tr is the sum of the trace of the DM. Second the DM must commute with the Hamiltonian:

$$[\rho, H] = 0 \tag{2.7}$$

Third, the DM must be idempotent:

$$\rho^2 = \rho \implies \rho(\mathbf{r}, \mathbf{r'}) = \int d\mathbf{r"} \rho(\mathbf{r}, \mathbf{r"}) \rho(\mathbf{r"}, \mathbf{r'}) \tag{2.8}$$

Idempotency of the DM is equivalent to the orthonormality of the Kohn-Sham

states in standard DFT, which is needed for a pure, fermionic, spin-collinear state, and to ensure that $f_n \in 0, 1$ From the density matrix the electron density can be obtained as:

$$n(\mathbf{r}) = 2\rho(\mathbf{r}, \mathbf{r}) \tag{2.9}$$

where the factor of two again accounts for spin degeneracy. The total energy of the non-interacting system E_s may be obtained via

$$E_s = 2Tr(\hat{\rho}\hat{H}) \tag{2.10}$$

consequently the energy of the real interacting system is calculated applying the double-counting corrections to the Hartree and exchange-correlation terms. The solution to equation 2.1 (i.e. Kohn-Sham equation) can then be obtained by minimizing the energy with respect to the density-matrix, subject to the constraints expressed in equations 2.6, 2.7 and 2.8.

Calculations based on the DM representation of the Kohn-Sham problem scales at best as N^2. This scaling is determined by two factors: the number of occupied states is directly proportional to N and each state extends over the whole system. Therefore in order to obtain a linear-scaling method, it is necessary to exploit the aforementioned nearsightedness of many-body quantum mechanics. In equation 4 the decay rate γ depends only on the energy gap between the highest occupied and lowest unoccupied states. Since the band gap is generally independent from the system-size the states described by the DM will not extend to the whole system anymore. This consideration opens to the possibility of imposing a spatial cutoff to the describe the states expressed in the DM. Therefore the total amount of significant information in the DM will only scales linearly with N.
In order to apply such spatial cutoff in ONETEP, the single-particle DM is recast in a separable form using atom-centered functions (Non-Orthogonal Generalized Wannier Functions, NWGFs [31]) φ_α as the basis set:

$$\rho(\mathbf{r}, \mathbf{r'}) = \sum_{\alpha,\beta} |\varphi_\alpha\rangle K^{\alpha\beta} \langle\varphi_\beta| \tag{2.11}$$

where $K^{\alpha\beta}$ is known as density kernel [32]. $K^{\alpha\beta}$ is the representation of the density-matrix in the set of duals of the NGWFs defined by:

$$\langle\varphi_\beta|\varphi^\alpha\rangle = \int d^3r \varphi^*_\beta(\mathbf{r})\varphi^\alpha(\mathbf{r}) = \delta^\alpha_\beta \tag{2.12}$$

$K^{\alpha\beta}$ has elements which are nonzero only if $|r_\alpha - r_\beta| < r_c$ (due to nearsightedness effects), where r_α and r_β indicate the coordinates of the centers of the NGWFs φ_α and φ_β, and r_c is the imposed real space cutoff length. Imposing spatial cut-offs on the NGWFs and the density-kernel results in a density-matrix whose information content scales linearly with system-size, an approximation which is controlled by adjusting the r_α and r_c. In practice, these cut-offs are increased until the desired physical properties of the system converge.

2.1.3 Density Kernel and NGWFs Optimization

Linear-scaling methods mainly fall into two categories. The first approach optimizes the density-kernel for a fixed, but large, set of local orbitals [33, 34, 35] while the second involves the optimization in situ of both the density-kernel and basis-set[36, 37]. ONETEP adopt the latter approach and performs both density-kernel and NGWF optimization self-consistently in similar fashion to what happens in ensemble DFT method for metallic systems [38].

The ground state of the Khon-Sham problem recast in DM terms can be reached through optimization of the DM itself while imposing the constrains in equations 2.6, 2.7, 2.8. In ONETEP this optimization procedure can be expressed as:

$$\begin{aligned} E^0 = \min_n \; E[n] \; = \min_\rho \; E[\rho] \; = \min_\rho \; E[\rho(\{\mathbf{K}^{\alpha\beta}\}, \{\varphi_\alpha\})] \; = \\ \min_{\{\mathbf{K}^{\alpha\beta}\},\{C_{KLM,\alpha}\}} \; E[\rho(\{\mathbf{K}^{\alpha\beta}\}, \{C_{KLM,\alpha}\})] \; = \min_{\{C_{KLM,\alpha}\}} \; \epsilon[\{C_{KLM,\alpha}\}] \end{aligned} \tag{2.13}$$

with:

$$\epsilon[\{C_{KLM,\alpha}\}] = \min_{\{K^{\alpha\beta}\}} \; E[(\{\mathbf{K}^{\alpha\beta}\}; \{C_{KLM,\alpha}\})] \tag{2.14}$$

Equation 2.13 and 2.14 define the nested loops (one for the NGWFs, φ_α, and one for the density kernel, $\mathbf{K}^{\alpha\beta}$) strategy used to minimize the DFT-energy. In the outer NWGFs-optimization loop, the density kernel is kept fixed while the NGWFs coefficients (i.e. $C_{KLM,\alpha}$) are variationally optimized to reach the minimum self-consistent energy. Once the NGWFs optimal coefficients are updated, the total energy is minimized in the inner cycle with respect to the elements of $\mathbf{K}^{\alpha\beta}$, while the NGWFs coefficients are kept fixed. This alternated cycle is repeated until self-consistency of the DFT-energy with respect to both the variational parameters (the expansion coefficients for φ_α and $\mathbf{K}^{\alpha\beta}$) is achieved.

The $\mathbf{K}^{\alpha\beta}$ optimization cycle is based on algorithms that minimize the total energy while driving the DM towards idempotency. These schemes make use of a penalty term[39] to the energy functional, dependent on DM-idempotency itself.

The basic idea at the roots of these penalty-functional methods is the purification scheme proposed by McWeeny[32]:

$$\tilde{E} = E[\rho] + \alpha P[\rho] \tag{2.15}$$

with:

$$P[\rho] = Tr[(\rho - \rho^2)^2] \tag{2.16}$$

Close to idempotency, the steepest descent optimization of P results in an iterative algorithm that increasingly improve a trial DM:

$$\rho(\mathbf{r}, \mathbf{r'})_{k+1} = 3\rho(\mathbf{r}, \mathbf{r'})_k^2 - 2\rho(\mathbf{r}, \mathbf{r'})_k^3 \tag{2.17}$$

McWeeny purification scheme is proved to converge when the initial occupancies of the trial DM lie in the $\left(\frac{1-\sqrt{5}}{2}, \frac{1+\sqrt{5}}{2}\right)$ interval. When tighter bounds are considered and the occupancies lie in the $\left[-\frac{1}{2}, \frac{3}{2}\right]$ range, then the ρ_{k+1} DM (called purified DM) will be weakly idempotent with all its entries ranging from $[0, 1]$ range. ONETEP employs McWeeny purification scheme as the first step in the DFT-energy minimization. Once an acceptable level of idempotency has been reached, ONETEP switch to a variant of the Li-Nunes-Vanderbilt purification scheme[40]: the Haynes, Skylaris, Mostofiand Payne (HSMP) one [41]. In this variant, an auxiliary matrix σ is defined in terms of the purified and renormalized DM:

$$\rho = N\frac{3\sigma^2 - 2\sigma^3}{Tr\left[3\sigma^2 - 2\sigma^3\right]} \tag{2.18}$$

The denominator in equation 2.18 gives rise to terms in the search direction that project out the electron number gradient by construction self-determining in this way the chemical potential of the system. When the density kernel converges to the desired tolerance, the NGWFs are updated (in an outer optimization cycle). NGWFs are support functions derived from a subspace rotation $\mathbf{M}$ of the Wannier Function [42]:

$$|\varphi_{\alpha\mathbf{R}}(\mathbf{r})\rangle = \frac{V}{(2\pi)^3}\int_{1_{st}BZ} dk e^{-ik\cdot R}\sum_n |\Psi_{nk}(\mathbf{r})\rangle \left[\mathbf{M}_n^{\dagger\beta}\mathbf{S}_{\alpha\beta}\right] dk \tag{2.19}$$

Where:

$$\mathbf{M}_n^{\alpha}(\mathbf{k}) = \int d\mathbf{r}\varphi_{\alpha\mathbf{R}}(\mathbf{r})\Psi_{nk}(\mathbf{r}) \tag{2.20}$$

is the transformation matrix and $\Psi_{nk}(\mathbf{r})$ and is a pseudo-atomic orbital used as starting guess for the initial NGWFs representation.
To perform the *in-situ* basis set optimization, it is necessary to express the NGWFs in some underlying set of primitive functions. ONETEP, relies on periodic cardinal sine (psinc) functions [43]. An example of these function behavior is reported in Fig. 2.1. Each one of these psinc functions is centered on a grid point of a regular mesh commensurate with the simulation cell.

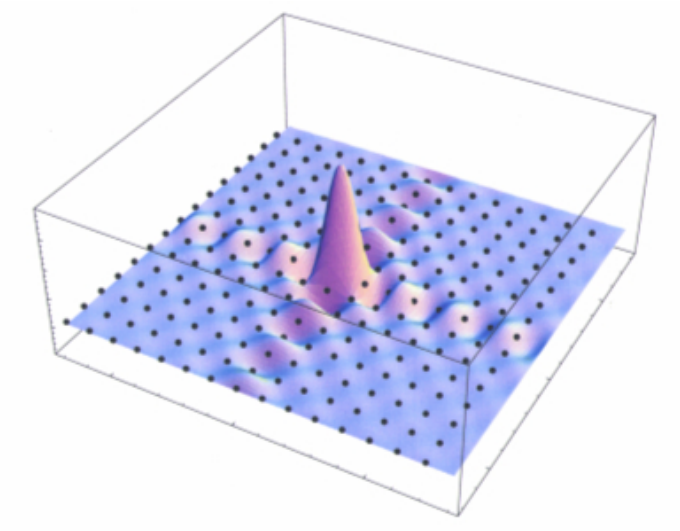


Figure 2.1: A psinc basis function used to expand the NGWFs in ONETEP.

The NGWFs expansion in term of this basis reads:

$$|\varphi_{\alpha\mathbf{R}}(\mathbf{r})\rangle = \sum_{K=0}^{N_1-1}\sum_{L=0}^{N_2-1}\sum_{M=0}^{N_3-1} C_{KLM,\alpha} D_{KLM}(\mathbf{r}) \tag{2.21}$$

where $C_{KLM,\alpha}$ are the effective expansion coefficients optimized in equation 2.13 $D_{KLM}(\mathbf{r})$ is the psinc function defined as:

$$D_{KLM}(\mathbf{r}) = \frac{1}{N_1 N_2 N_3} \sum_{P=-J_1}^{J_1}\sum_{Q=-J_2}^{J_2}\sum_{R=-J_3}^{J_3} e^{i(P\mathbf{B}_1+Q\mathbf{B}_2+R\mathbf{B}_3)\cdot(\mathbf{r}-\mathbf{r}_{KLM})} \tag{2.22}$$

Psinc functions are formed from a discrete sum of plane-waves, which makes them independent from the nuclear coordinates and systematically improvable upon increase of the kinetic energy cut-off, retaining in this way the positive aspects of the plane-waves. They also allow for accurate and efficient calculation of the kinetic

energy via Fast Fourier Transforms (FFTs) and are orthogonal by construction [44]. The FFTs used to calculate the kinetic energy however involve global operations (i.e. all the system has to be considered). This operation makes the time to calculate the kinetic energy of a single NGWF scale as O(N log N), resulting in an overall scaling higher than N^2.
This issue is solved through a very innovative strategy, known as FFTbox method[45]; such expedient permits ONETEP to reach true linear scaling. This strategy rests on the localization of the NGWFs in real space. Their localization permits evaluation of the various DFT Hamiltonian matrix elements in a cell (the FFTbox) smaller than the whole simulation cells. Provided the localization radius of the NWGFs is kept constant, the size of the FFTbox can be left unchanged as the size of the simulation-cell increases (Fig. 2.2). In this way the cost of integral evaluation becomes independent of the overall system size and allows for highly parallel linear scaling execution of DFT simulations. The FFT boxes are chosen to be sufficiently large to include the central NGWF and all of its overlapping neighbors.

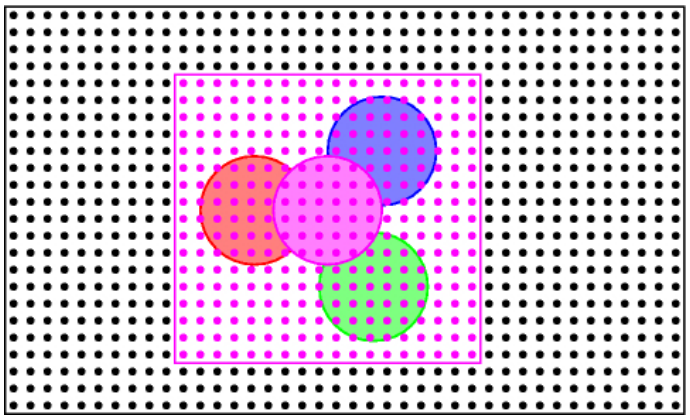


Figure 2.2: Illustration of the FFT box technique used in ONETEP.

In ONETEP the FFTbox technique is employed to calculate the kinetic energy, to interpolate the NGWFs before the calculation of the density and to evaluate every term in the Hamiltonian [46]. In this way making use of the FFTbox scheme ONETEP evaluates and minimize the energy of the system with respect to the DM while keeping the density kernel fixed and optimizing the psinc function coefficient of each NGWFs.

2.2 Path-Integral Molecular Dynamics

Feynman's path-integral formulation[47] of quantum statistical mechanics makes possible the computer simulation of quantum many-body systems of chemical and

physical interest[47, 48]. These types of simulations have had considerable impact on our ability to analyze the properties of quantum many-body systems at finite temperature.

Path-Integral techniques are computationally tractable[10, 49] and perfectly suited for implementation on modern day parallel computing architecture. Methods exploiting the path integral formulation involve the calculation of many-dimension integrals in order to evaluate numerous equilibrium observable properties, including thermodynamic and structural quantities. As mentioned in the previous chapter, only the important aspect of the theory behind Path-Integral Molecular dynamics would be discussed below.

Considering a single-particle system, having mass, m, momentum, p, and potential, $\phi(x)$ as described by the Hamiltonian:

$$H = \frac{p^2}{2m} + \phi(x) \equiv K + \Phi \tag{2.23}$$

the canonical partition function $Z(\beta)$, according to Feynman's formulation [47], is the trace of the density matrix ρ and can be evaluated in the coordinate basis. The coordinate basis are continuous, as such, $Z(\beta)$ can be expressed as an integral:

$$\rho = e^{-\beta H} \tag{2.24}$$

$$Z(\beta) = Tr(e^{-\beta H}) = \int dx \langle x|e^{-\beta(K+\Phi)}|x\rangle \tag{2.25}$$

where K and Φ are the kinetic and potential operators respectively. The operators K and Φ do not commute so the exponential, $e^{-\beta(K+\Phi)}$, cannot be evaluated directly. However, this limit can be circumvented by exploiting the Trotter theorem[50] which allows the partition function $Z(\beta)$ to be expressed as a product of P factors of the operator, Ω as shown in equation 2.26. Ω is a define operator, $\Omega = e^{-\frac{\beta}{2P}\Phi} e^{-\frac{\beta}{P}K} e^{-\frac{\beta}{2P}\Phi}$.

$$Z(\beta) = \lim_{P\to\infty} \int dx \langle x|\Omega^P|x\rangle = \lim_{P\to\infty} \int dx \langle x|\Omega.\Omega...\Omega|x\rangle \tag{2.26}$$

The full derivation with the Trotter theorem can be found in reference [51]. The coordinate space matrix elements of Ω, is then evaluated considering the completeness relation for momentum eigenstates. This then makes the canonical partition function to be re-written as:

$$Z_P(\beta) = \left(\frac{mP}{2\pi\beta\hbar^2}\right)^{P/2} \int dx_1...dxp \quad exp\Bigg\{ -\sum_{i=1}^{P} \left[\frac{mP}{2\beta\hbar^2}(x_{i+1} - x_i)^2 + \frac{\beta}{P}\phi(x_i)\right]\Bigg\}_{x_{P+1}=x_1} \tag{2.27}$$

A chain frequency, $\omega p = \sqrt{\frac{P}{\beta\hbar}}$, and an effective potential is introduced into equation 2.27 which maps the quantum particle to a set of P-classical particles. This allows the quantum partition function to be re-written as a classical configuration partition function for a P-particle system shown in equation 2.28.

$$U_{eff}(x_1, ..., xp) = \sum_{i=1}^{P} \left[\frac{1}{2} m\omega^2 p(x_{i+1} - x_i)^2 + \frac{1}{P}\phi(x_i) \right]_{x_{p+1}=x_1} \tag{2.28}$$

$$Z_P(\beta) = \left(\frac{mP}{2\pi\beta\hbar^2} \right)^{P/2} \int dx_1 ... dx_P \quad e^{-\beta U_{eff}(x_1,...x_P)} \tag{2.29}$$

The classical particles are connected together by a harmonic spring. The mapping of the quantum system to P-particle fictitious system is know as *classical isomorphism*. This is illustrated in Figure 2.3. Due to the path resembling a necklace, the classical particles are sometimes referred to as *"beads"*.

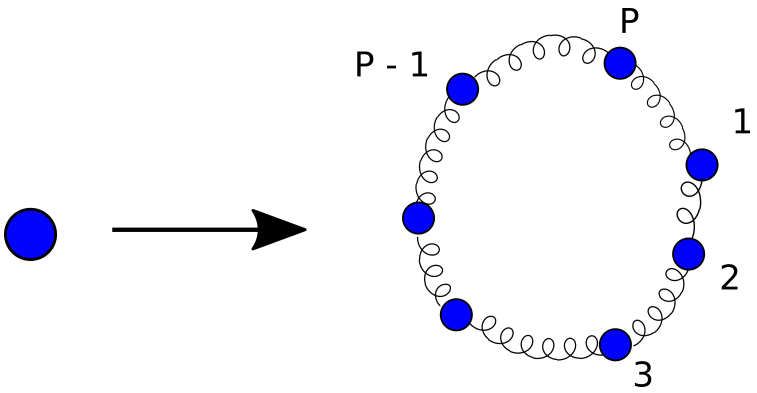


Figure 2.3: Illustration of *classical isomorphism*: The connection between a single quantum particle and the discretized P-particle system.

By introducing momenta into equation 2.29, the quantum partition function is evaluated according to the classical Hamiltonian shown in equation 2.31 :

$$H(p, x) = \sum_{i=1}^{P} \frac{p_i^2}{2\tilde{m}_i} + U_{eff(x_1,...x_p)} \tag{2.30}$$

$$H(p, x) = \sum_{i=1}^{P} \left[\frac{p_i^2}{2\tilde{m}_i} + \frac{1}{2} m\omega^2 p(x_{i+1} - x_i)^2 + \frac{1}{P}\phi(x_i) \right] \tag{2.31}$$

The quantum partition function is exactly reproduced in the limit of $P \to \infty$.

Previous studies[7, 52, 53] with PIMD have shown that 32 beads, leads to a convergence in structural properties but the simulations are computationally expensive. However, recent developments have allowed PIMD simulations to be done with much fewer number of beads e.g. (2,4,6 ..)[3, 14] instead of 32. This is achieved by the use of sophisticated thermostats combined with PIMD. There are several variations of these thermostats based on the Generalized Langevin Equation(GLE) which are implemented in i-PI. These include methods like Path-Integral + GLE[54] and PIGLET[14]. The latter ensures convergence of both the potential energy and the quantum kinetic energy [54, 55]. The essential idea here is to apply a classical thermostat to the centroid and a GLE thermostat to the internal modes of the ring polymer.

Chapter 3

Implementation: i-PI-ONETEP Interface

3.1 Project Design:Server-Client Model

The aim of this project is to achieve petascale linear scaling (LS) scientific computation on extended systems constituting over 10,000 atoms. These scientific computations include classical molecular dynamics, path-Integral molecular dynamics, parallel tempering simulations and replica exchange molecular dynamics. This Chapter describes the project design and worki flow required to attained the intended goal.

The two codes i-PI[3] and ONETEP [9], will provide the basis to tackle this computational challenge. In order to obtain maximal performance from the integration of these codes, a server-client scheme is the best fit. The i-Pi [3] code which runs as a server will account for the numerical integration and the evolution of the nuclear degrees of freedom according to the equation of motion. ONETEP will acts as the client, calculate and send to i-PI and the required0 quantities which includes forces($\mathbf{f}$), the potential energy($\mathbf{U}$) and the virial(σ). This server-client model requires interfacing ONETEP, which is written in FORTRAN programming language with i-PI [3], which is written in PYTHON [56]. The synergistic effect of i-PI and ONETEP is afforded by a driver subroutine, ***driver_ipi***. Figure 3.1 shows a schematic overview of the project design.

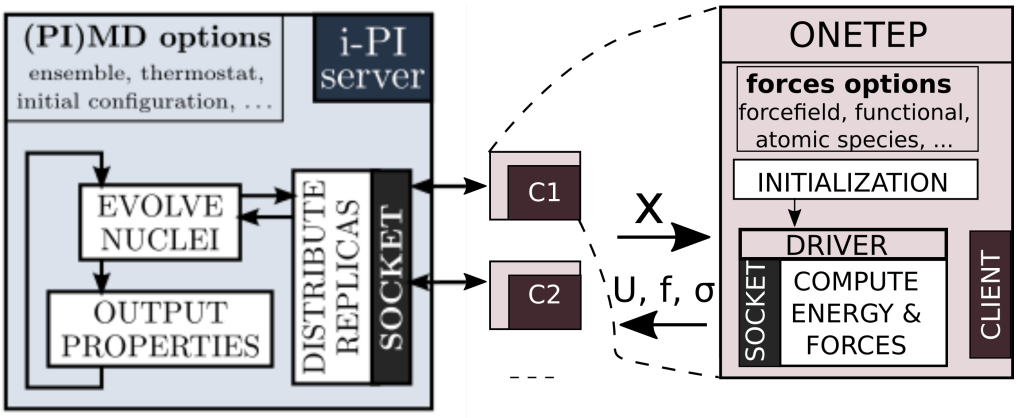


Figure 3.1: Project Design

The ***driver_ipi*** subroutine controls the operations at any given instance by using string objects to determine the state of both the server and the client in order to determine the scientific computation that is needed to be done. The design of i-PI requires the server-client interconnection to be via a socket protocol which is a module in the C programming language. This makes the server-client model a heterogeneous code. Thus, in order to accomplish an efficient exchange of data between codes of different programming languages, interoperability of codes is required. The client, ONETEP, written in FORTRAN, interoperates with C-programming language.

Different programming languages on various compilers and various platforms have different representation of data, especially floating point types. To ensure proper interoperability between the FORTRAN and C objects, data types have to be asserted and this requires the use of $ISO\_C\_BINDING$ [1]. $ISO\_C\_BINDING$ is an intrinsic module in FORTRAN that is used to guarantee that FORTRAN and C objects agree on data types. The socket protocol establishes communication between the two codes in order to transfer and receive numerical quantities of interest in a communication cycle.

A communication cycle is determined by the nature of the computation to be carried out in order to achieve the desired scientific goal. In a classical molecular dynamics calculation, the number of communication cycles is determined by the number of molecular dynamics steps. i-PI (as a server), initiates the cycle by opening a specified port and listen on it for incoming connections from clients, in this case, ONETEP. The electronic structure parameters are read from ONETEP input data and the atomic coordinates and other parameters of the calculation intended to be done regrading the nuclei positions evolution, owing to the benefit

[1]ISO_C_BINDING was added to the FORTRAN language from the 2003 definition onwards but some compilers let FORTRAN90 codes access it.

of decoupled force and nuclear evaluation, are provided by i-PI data input. Once a server-client connection has been established, the driver code uses string objects to communicate, first, the status of both server and client before data transfer and computation commences. i-PI sends to ONETEP the dimensions of the simulation box, specifically, the lattice vectors, reciprocal lattice vectors($\mathbf{h}$) and the atomic positions of the nuclei of the system of study($\mathbf{x}$).

The socket protocol allows i-PI to have multiple clients which enables it to take advantage of having ionic forces computed in an embarrassingly-parallel fashion. This is both essential and efficient for simulations such as path-integral molecular dynamics where ionic forces of multiple atomic coordinate replicas have to be evaluated[2]. When the system coordinates are received, ONETEP computes and returns the ionic forces($\boldsymbol{f}$), electronic energy($\boldsymbol{U}$), virial tensor(σ), dipole moment, partial charges of the atoms and any other quantity of interest derivable by an *ab-initio* DFT SCF calculation.

3.2 Communication Protocol: Socket

The socket protocol relies on the exploitation of the transport layer in the Open Systems Interconnection (OSI) model to establish communication between two different processes. This protocol makes available two possible connection channels, a UNIX and an Internet socket. The UNIX socket allows communication between processes on the same network and nodes. This allows faster transfer of data, hence, it is very useful in the instance of a shared memory paradigm computation. In contrast to the UNIX socket, the Internet socket allows for communication between processes on different networks. This makes it possible to exploit a server-client communication in distributed computing and possibly between different HPC facilities. Wrappers for the socket connection is and data transmission are provided along with i-PI to aid the interfacing with ONETEP. The pseudo code in Algorithm 1 shows the core sections of the codes that are involved in the socket opening implementation using an Internet socket or a UNIX socket.

The output of a successful connection between i-PI and ONETEP is an integer value stored in the variable *socketID* as shown in the pseudocode. The value of *socketID* is an identifier of the respective established socket connection and also serves as a file descriptor. A file descriptor is a unique data handler between communicating processes which is subsequently used to transmit data.

The file descriptors are identified by the value of the variable *socketID* which is unique in each connected client, thus, each instance of ONETEP. This one-to-one communication identified by a unique numerical value of *socketID* is

[2]The cost in evaluating forces in a PIMD simulation is n times that of a classical simulation on the same data set(atomic coordinates) where n is the number of beads

what empowers i-PI to have multiple clients just as mentioned earlier in Section 3.1. The same unique *socketID* also allows i-PI to keep track of active instances of ONETEP. The *socketID* was used along with other intrinsic socket functions to implement wrappers for input and output streaming of data. This is shown in algorithm 2. After each instance of ONETEP establishes a connection with i-PI, a multiple-program-on-multiple-data(MPMD) technique of computing is performed to compute quantities in parallel.

Algorithm 1 Opening a socket in C

```
1: procedure OPENING_(INT SOCKETID, INT PORTNUMBER, CHAR * HOSTIPADDRESS)
Require: Int socketID
Require: Int PORTNUMBER
Require: Struct hostent * server
Require: Int socketSize
2:     if InternetSocket then
3:         STRUCT sockaddr_in &serv_addr
4:         SocketPointer ← (STRUCT sockaddr*)&serv_addr
5:         SocketSize ← sizeof(serv_addr)
6:         socketID ← socket(AF_INET, SOCK_STREAM, 0)
7:         server ← gethostbyname(hostIPAddress)
8:         ( ...set memory location to zero with bzero )
9:         ( ...copy server details with bcopy )
10:        serv_addr.sin_family ← AF_INET
11:        serv_add.sin_port ← htons(PORTNUMBER ) ▷ Convert port number network byte order
12:        connect(socketID,SocketPointer,SocketSize )
13:    else
14:        STRUCT sockaddr_un &serv_addr
15:        SocketPointer ← (STRUCT sockaddr*)&serv_addr
16:        SocketSize ← sizeof(serv_addr)
17:        socketID ← socket(AF_UNIX, SOCK_STREAM, 0)
18:        ( ..set memory location to zero with bzero )
19:        ( ...copy server details with bcopy )
20:        serv_addr.sin_family ← AF_UNIX
21:        (...get host details .. )
22:        connect(socketID,SocketPointer,SocketSize )
23:    end if
24: end procedure
```

3.3 Driver Module

The socket implementation described above is used to develop a module which contains the subroutine, *driver_ipi* whose purpose is to drive the entire working cycle. The driver subroutine uses string objects as keywords to determine the required computation to be performed at any instance just as mentioned in Section 3.1. The keywords include ”STATUS”, ”NEEDINIT”, ”HAVEDATA”, ”READY”, ”POSDATA”, ”GETFORCES” and ”FORCEREADY”. Each of theses keywords play crucial roles that aids the computations in each communication cycle. Their roles are illustrated on the flow chart in Figure 3.2 as well as in the pseudocode shown in Algorithm 3.

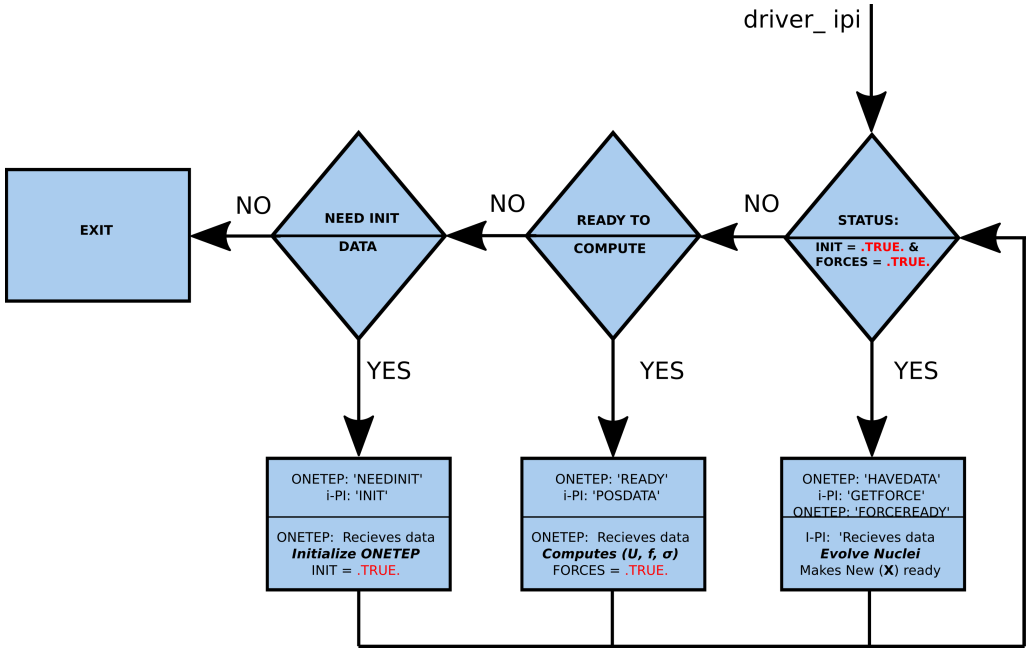


Figure 3.2: Work flow

The server code, i-PI, runs as a daemon in the background whiles listening on a particular port, known apriori by both server-client(the port number is set as a value to the variable *srvport* and its corresponding IP address parameter *srvaddress* in the ONETEP input). The driver uses the *host IP address* and *port* number to connect ONETEP to i-PI, which would be listening for incoming connections. This, then begins the inner cycle of receiving atomic coordinates, computing and sending inter-atomic forces and the needed quantities. The C-objects, from wrapper functions *readbuffer* and *writebuffer* are used by the root process of i-PI and that of ONETEP to read and write from or to each other. Algorithm 2 shows the structure of the C wrapper objects fro sending and receiving data.

Algorithm 2 Data Transfer Wrappers in C

```
1: procedure WRITEBUFFER_(INT * PSOCKETID, CHAR * DATA, INT * PDATASIZE)
Require: Int socketID
Require: Int datasize                          ▷ Pointer to data size in bytes
2:     socketID ← * psocketID
3:     datasize ← * pdatasize
4:     write(socketID, data, datasize)
5: end procedure

6: procedure READBUFFER_(INT *PSOCKETID, CHAR * DATA, INT * PDATASIZE )
Require: Int socketID
Require: Int datasize
7:     socketID ← * psocketID
8:     datasize ← * pdatasize
9:     read(socketID, data, datasize)
10: end procedure
```

As mentioned in Section 3.1, the socket programming structure is implemented in the C-programming language hence the BIND(C) module from ISO_C_BINDING is employed in the inter-interoperability of C-objects with FORTRAN objects to ensure and ascertain the expected data type as shown in Figure 3.3.

The pseudocode in Algorithm 3 shows the core section of the driver subroutine in ONETEP that is used to communicate with i-PI. When an instance of ONETEP(client) connects to i-PI(sever), the root process reads the transfered data using the *readbuffer* subroutine. It then broadcasts the received data to other processes within a given the MPI communicator. The data transfered by the root process are then assigned to their respective memory locations as illustrated in Algorithm 3. In each communication cycle, the amount of data transferred is expressed in the relation below where N is the number of atoms/particles.

$$\overbrace{\underbrace{(9 \times 8) \times 2}_{BOX\ dimensions} + \underbrace{(N \times 3 \times 8)}_{Atomic\ postions}}^{from\ i-PI\ to\ ONETEP} bytes \tag{3.1}$$

$$\underbrace{\underbrace{(N \times 3 \times 8)}_{Atomic\ positions} + \underbrace{(3 \times 9)}_{Virial\ tensor} + \underbrace{(1 \times 8)}_{Potential\ energy}}_{from\ ONETEP\ to\ i-PI} \quad bytes \tag{3.2}$$

Electronic structure calculations are then carried out to obtain forces and the other quantities of interest. The *writebuffer* is used to transfer the resulting data which consists of ionic forces, potential energy and virial tensor to i-PI to complete a cycle. The number of communication cycles following the depicted scheme executed is determined by the nature of the scientific investigation. In a molecular dynamics(MD) simulation, the number of communication cycle is determined by the number of molecular dynamic steps.

(a)

```
void open_socket_(int *psockfd, int* inet, int* port, char* host);
void writebuffer_(int *psockfd, char *data, int* plen);
void readbuffer_(int *psockfd, char *data, int* plen);
```

(b)

```
interface
  subroutine open_socket_(psockfd, inet, port,host)bind(C, NAME='open_socket')
     use iso_c_binding !! External dependency
     integer(c_int)::  psockfd, inet, port
     character(c_char), dimension(*):: host
  end subroutine open_socket_
end interface

interface
  subroutine  writebuffer_(psockfd, plen, data) bind(C, NAME='writebuffer')
     use iso_c_binding !! External dependency
     integer(c_int)::  psockfd, plen
     character(c_char) :: data
  end subroutine writebuffer_
end interface

interface
  subroutine  readbuffer_(psockfd, plen, data) bind(C, NAME='readbuffer')
     use iso_c_binding !! External dependency
     integer(c_int)::  psockfd, plen
     character(c_char) :: data
  end subroutine readbuffer_
end interface
```

Figure 3.3: Interoperability with C Objects

Algorithm 3 Driving Calculations on Extended Systems

1: **procedure** DRIVER_IPI(TYPE DEF MDL)
Require: Int port := 31415
Require: Character(LEN :=15) host
Require: Int socketID := 0
Require: character(LEN := 12) : MSGLEN ▷ size of string data in bytes
Require: Real(Kind=DP), Allocatable :: DATAbuffer(:)
Require: Real(Kind=DP), Allocatable :: atoms(:,:)
2: Call **open_socket**(socketID, inet, host, port)
3: **while do**
4: Call **readbuffer**(socketID, header, MSGLEN)
5: **if** *header* == "$STATUS$" **then**
6: **if** $ONETEP.not.Initialized$ **then**
7: Call **writebuffer**(socketID, "NEEDINIT", MSGLEN)
8: **else if** $hasdata(f, U, \sigma)$ **then**
9: Call writebuffer(socketID, "HAVEDATA", MSGLEN)
10: **else**
11: Call writebuffer(socketID, "READY", MSGLEN) ▷ Initialized and Ready to compute forces ...
12: **end if**
13: **else if** *header* == "$INIT$" **then**
14: Call readbuffer($socketID, NAT\_Vec, MSGLEN$)
15: Call (...Allocate memory for atomic positions ...)
16: Call readbuffer($socketID, DATAbuffer, N$)
17: (... initialize ONETEP ...)
18: $init \leftarrow .true.$
19: **else if** *header* == "$POSDATA$" **then**
20: Call readbuffer($socketID, lattice\_Vec, MSGLEN$)
21: Call readbuffer($socketID, Recip\_lattice\_Vec, MSGLEN$)
22: Call readbuffer($socketID, nat, MSGLEN$) ▷ No. of atoms
23: allocate(DATAbuffer(3,nat)) ▷ Allocate memory loc. for communication
24: Call readbuffer($socketID, DATAbuffer, nat * 3 * 8$) ▷ No. of atoms
25: allocate(atoms(nat,3)) ▷ Allocate memory to contain atomic coordinates
26: **for all** $i \in nat$ **do**
27: $atoms(i, :)$:= *DATAbuffer(3*(i-1)+1 : 3 *i)* ▷ Re-Packing coordinates
28: **end for**
29: $mdl\%atomicCoord \leftarrow atoms$
30: $mdl\%latcell \leftarrow TRANSPOSE(cell\_h)$
31: $mdl\%Reciplatcell \leftarrow TRANSPOSE(cell\_ih)$
32: CALL $energy\_force\_calculate$(potenergy, forces, mdl) ▷ Compute forces ...
33: CALL **virial**($virial\_tensor, forces, mdl$)
34: $hasdata \leftarrow .true.$

```
35:         else if header == "GETFORCE" then
36:             Call writebuffer(socketID, "FORCEREADY", MSGLEN)
37:             Call writebuffer(socketID, potenergy, 8)
38:             Call writebuffer(socketID, combuf, nat*3*8)
39:             Call writebuffer(socketID, virial_tensor, nat * 3 * 8)
40:             Call writebuffer(socketID, "NOTHING", 7)
41:             hasdata ← .false.
42:         else
43:             EXIT
44:         end if
45:     end while
46: end procedure
```

3.4 Serialized Evaluation of Quantities

In the instance of performing simulations such as classical *ab initio* molecular dynamics , the electronic structure part of the problem is performed in serialized steps(molecular dynamic steps.) This involves a connection between i-PI and a single instance of ONETEP where the driver dictates the work flow, as described in Figure 3.2, to evaluate $\phi(x)$. However, in simulations involving multiple replica such as PIMD, the registered instance of ONETEP will continually evaluate the needed quantities of each bead, $\phi(x_i)$ in the P-particle classical system for each MD step. This increase the time to solution of such simulations by a factor of the number of replicas(beads), P.

3.5 Parallel Evaluation of Quantities

AS mentioned earlier in Chapter 2, in simulations such as PIMD, each bead is separately subject to the external potential, ϕ. So ϕ, as shown in equation 2.31 from Chapter 2 and also below, can be evaluated in parallel.

$$H(p,x) = \sum_{i=1}^{P} \left[\frac{p_i^2}{2\tilde{m}_i} + \frac{1}{2} m\omega^2 p (x_{i+1} - x_i)^2 + \frac{1}{P}\phi(x_i) \right] \tag{3.3}$$

Since the design of the socket protocol allows i-PI to connect to multiple clients, each connected ONETEP instance can then evaluate the required quantities of

each bead, $\phi(x_i)$. The calculations are performed with a multiple-program-multiple-data(MPMD)[3] technique rather than a Single instruction multiple(SIMD) data technique. To perform the calculation in a SIMD fashion requires major changes to ONETEP which is quite involving.

[3]It is also referred to as multiple instruction multiple data(MIMD)

Chapter 4

Tests and Benchmarks

This chapter presents some preliminary tests results obtained from performing some MD simulations with the i-PI-ONETEP implementation. We also present some scalability benchmarks from the implementation.

A water dimer system, $(H_2O)2$, consisting of four hydrogen(H) and two oxygen(O) atoms was used for the test. We performed three different types of simulations, specifically for these benchmarks. The input parameters were taken from the tutorial package provided as part of the i-PI distribution.[1]

1. Molecular dynamics with 1 bead
2. Path-Integral molecular dynamics with 2 beads
3. Path-Integral molecular dynamics with 6 beads

In all of these runs, the *pile_g* thermostat was used. The *pile_g* thermostat attaches a white noise langevin thermostat to the normal mode representation[3]. Table 4.1 shows the parameters for both the electronic structure part from ONETEP and the molecular dynamics part of the simulation from i-PI.

[1] The parameters are from .xml input file found in the tutorial package provided with the i-PI distribution. Path: i-PI_TOP_DIRECTORY/examples/tutorial/tutorial-1.

Table 4.1: *Parameters for test MD simulations*

ONETEP		*i-PI*	
Parameter	*Value*	*Parameter*	*Value*
Pseduopotential	Norm conserving	Ensemble	NVT
psinc kinetic enr cutoff	500 eV	#Beads	(2 & 6)
O #NGWF	4	Temperature	25 K
O NGWF cutoff	10.0 bohr	Timestep	1.0 femtosec
H #NGWF	1	Thermostat	pile_g
H NGWF cutoff	8.0 bohr	Box Dim	20 X 20 X 20
Exchange functional	BLYP		
Convergence criterion	1.E-5 eV		

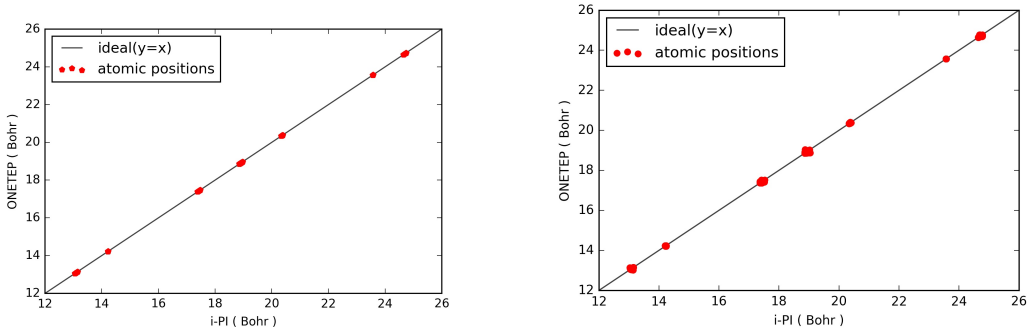
4.1 Communication Testing

The Classical and path-integral MD simulation respectively presents a peculiar use of the socket communication protocol regarding data transfer. Analysis of the trajectories, ionic forces and pressure tensor has been made in order to verify the proper functioning of the communication protocol. These analysis were made after 5 MD steps.

Figure 4.1 shows the results of the communication test where data is being sent from i-PI to ONETEP in the MD simulations. The atomic coordinates(X) sent from i-PI was cross-referenced with that received by ONETEP. The data points, representing the atomic coordinates of the water dimer system, on the linear curve $y = x$ shows that there's a match in atomic positions on both sides of the communication channel, in both the classical MD, Figure 4.1a and the path-integral MD simulation Figure 4.1b.

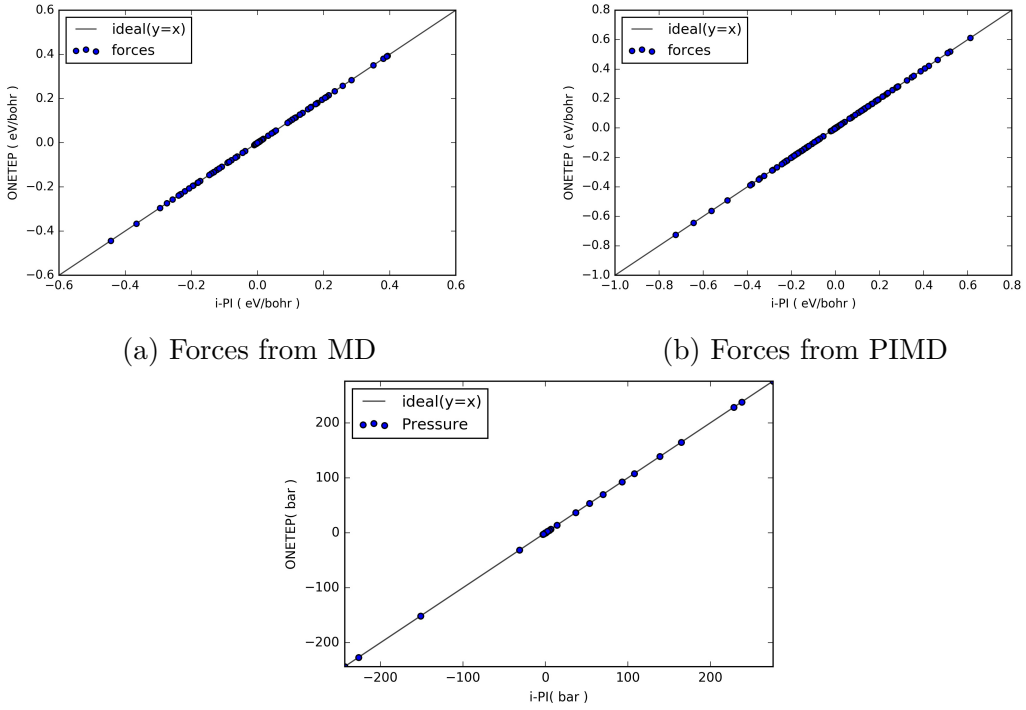
Figure 4.2 shows the results of the communication test where data is being sent from ONETEP to i-PI in both classical MD and path-integral MD simulations. The data point, represents the computed ionic forces in both classical MD, Figure 4.2a, and path-integral MD, Figure4.2b and the pressure tensor in a classical MD 4.2c simulation. The data point of the forces and the virial tensor on the curve $y = x$ indicate a data-match when cross-referenced. The match shows that data is being transfered from ONETEP to i-PI successfully.

The successful transfer of data through the channel protocol shows that the socket protocol functions properly.



(a) Atomic coordinate from MD (b) Atomic coordinate from PIMD

Figure 4.1: Communication check between i-PI and ONETEP from the transfer of Atomic positions, ($\mathbf{X}$), from i-PI to ONETEP in both Classical and Path-Integral Molecular dynamics.



(a) Forces from MD (b) Forces from PIMD

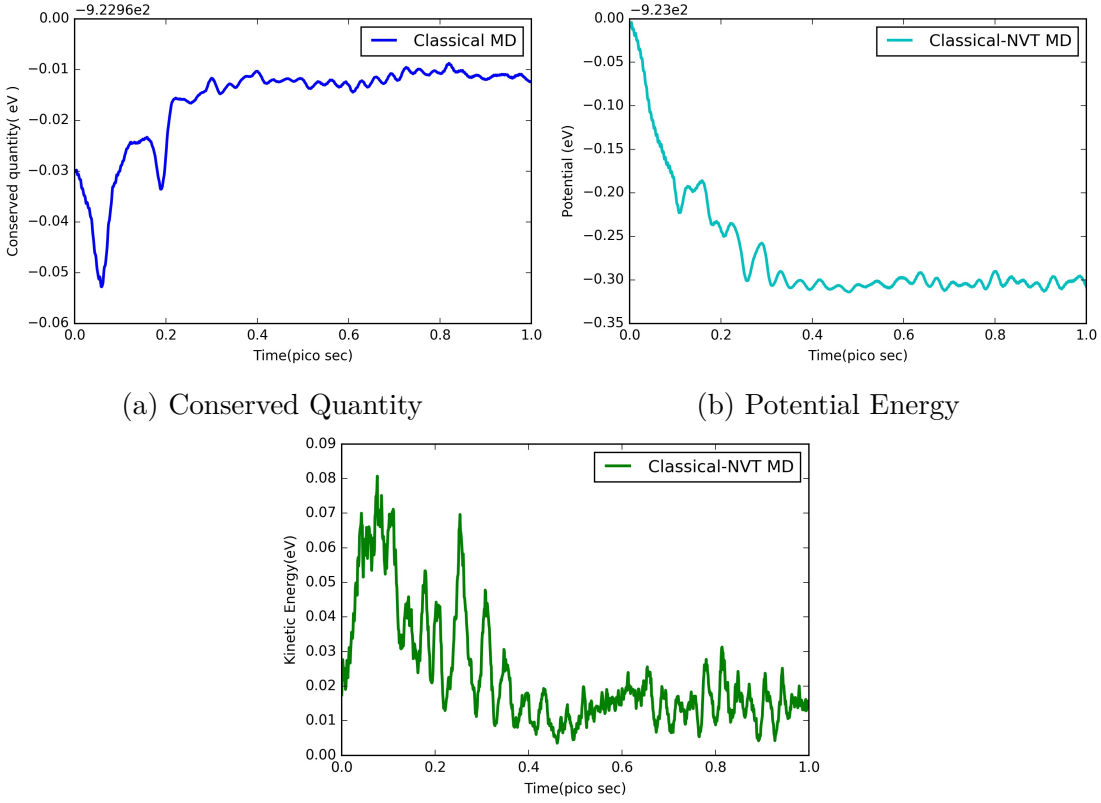
(c) Pressure Tensor From MD

Figure 4.2: Communication check between ONETEP from the transfer of ionic forces, ($\mathbf{f}$), and virial tensor, σ, from ONETEP to i-PI in both Classical and Path-Integral Molecular dynamics.

4.2 Preliminary Results of Simulations

In this section, we show the thermodynamics and structural properties of the water dimer system stemming from the Classical MD and PI simulations.

Figure 4.3 shows plots of the conserved quantity, the potential energy and the kinetic energy as function of time from the classical MD simulation. It can be observed from the plots that the system goes into equilibrium in 0.4 picoseconds with small fluctuations in the potential and kinetics energy. Also, the temperature is kept at 25 Kelvin. A plot of the temperature is shown in Figure A.5 in the Appendix.



(a) Conserved Quantity

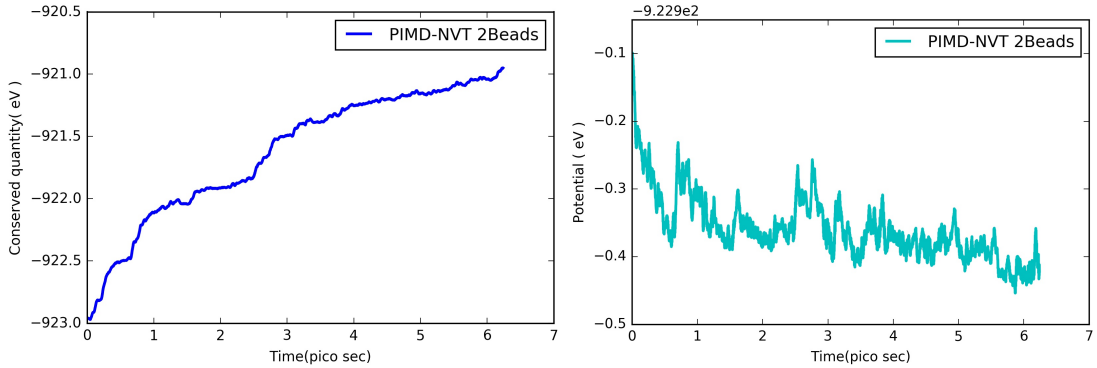
(b) Potential Energy

(c) Kinetic Energy

Figure 4.3: Plots of water dimer system properties from a Classical MD simulation

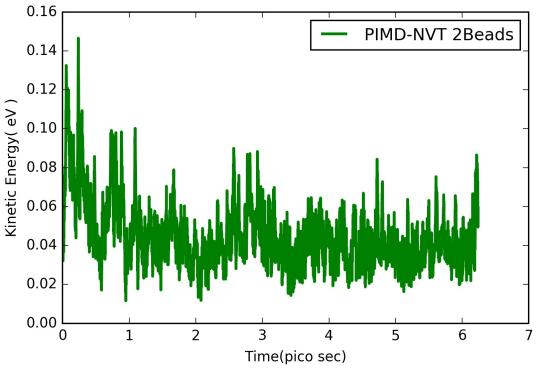
Figure 4.4 shows thermodynamic properties of path-integral simulations with two beads and six beads. Both simulation are still equilibrating even at the run time of 6 and 1.2 picoseconds respectively. Even though there are no large fluctuations in the kinetic and potential energies, the conserved quantities are drifting. This could be attributed to the time step being 1 femtoseconds, being

used in the simulation and possibly the convergence criterion for the computation of the ionic forces.



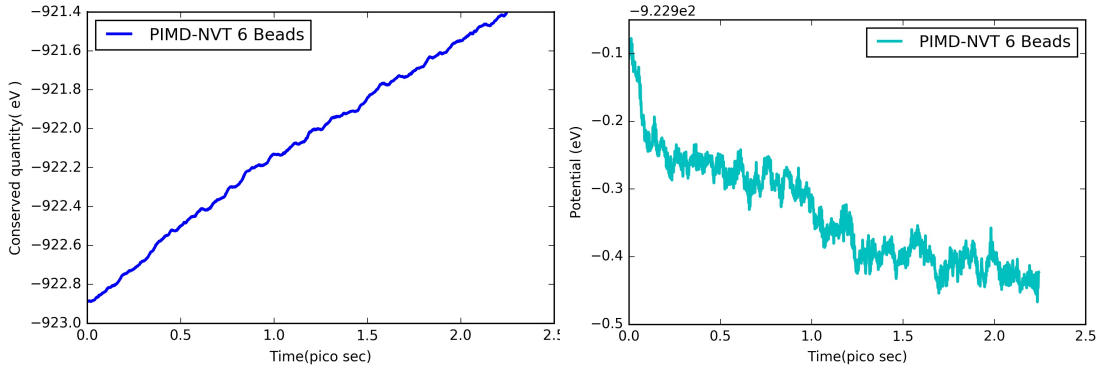
(a) Conserved Quantity

(b) Potential Energy

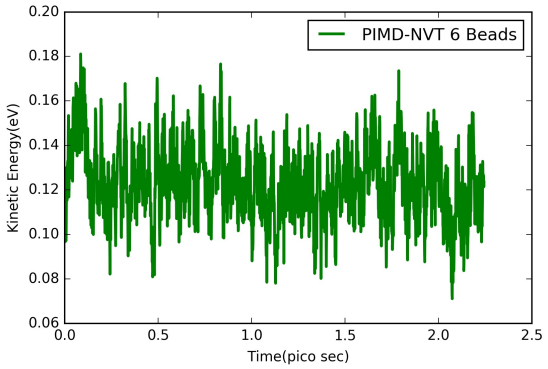


(c) Kinetic Energy

Figure 4.4: Plots of water dimer system properties from a PI MD simulation with 2 beads



(a) Path-Integral MD: Conserved Quantity (b) Path-Integral MD: Potential Energy



(c) Kinetic Energy

Figure 4.5: Plots of water dimer system properties from a PI MD simulation with 6 beads

Figure 4.6 shows the O-H radial distribution function(RDF) from the three different simulations. The RDF was generated from the entire trajectory from the simulation. The effect of quantum fluctuations from the PI simulations is shown in the broadening of the RDF of the hydrogen bond when compared to that of the classical simulation. These preliminary results shows that ONETEP has been successful interfaced with i-PI. Further tests with different systems are being carried out to ensure the proper functioning the ONETEP-iPI interface and to explore it's full capabilities.

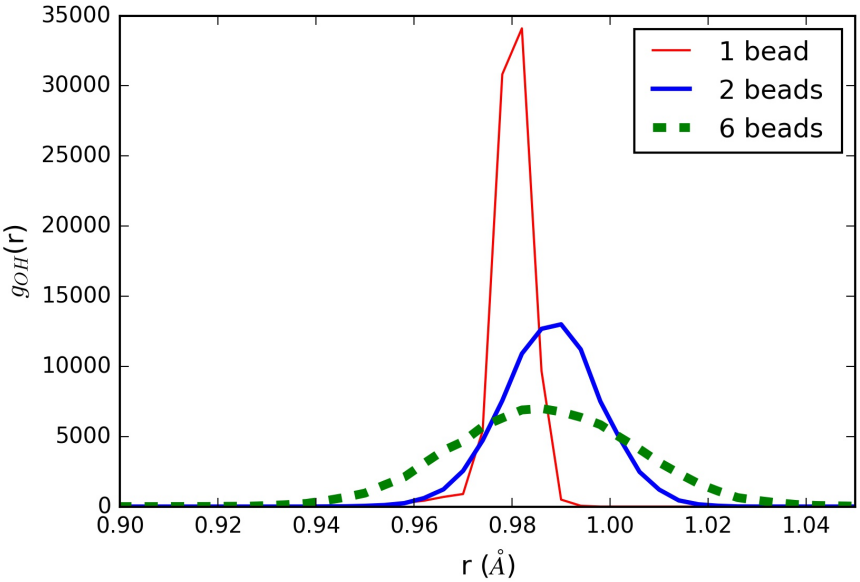


Figure 4.6: The OH Radial Distribution function in water dimer from NVT classical MD and PI simulations at 25 K.

4.3 Scalability: Porting to Intel's Knight Landing(KNL)

This section focuses on the results obtained testing the i-PI ONETEP interface on a Tier-0 machine. These tests should, in principle, highlight the advantage of the embarrassingly parallel implementation of our interface and confirm the viability of *ab-initio* path integral Peta-scale simulations.

All the benchmarks calculations were performed on the KNL A2 partition of CINECA' s Tier-0 system: Marconi. This partition counts 3600 nodes each one formed by 68 cores for a total of 244.800 cores in total, resulting in a peak performance of 11 PFlop/s. Each KNL node consists of four quadrants with respectively 18, 18, 16 and 16 cores.

The system picked to test the strong scaling behavior of our interface was water slab comprised of 2306 atoms. The size of the system was chosen in order to target what could be a prototypical calculation running just below the peta-scale regime (considering 6 beads in the PIMD dynamic). The calculations were run using respectively 4 and 1 NGWFs for the O and H atoms. A cutoff radius of 10.0 bohr was employed for the oxygen atoms while 8.0 bohr were used for the hydrogen ones. The adopted kinetic cutoff was of 800 eV; in all cases no truncation of the density kernel was enforced. In all the simulations, separable (Kleinman-Bylander) norm-conserving pseudopotentials [57], constructed with the opium code, were chosen.

All the simulations were performed with periodic boundary condition with a repetitive unit cell measuring 156.00 x 85.00 x 15.64 Å. The box dimensions on the y-axis ensured at least 15 Å of vacuum between replicated images in the non-periodic direction. The benchmark calculations for strong scaling were run starting from the smallest possible fitting unit (2 KNL = 136 cores) reaching up to 32 nodes (i.e 2176 cores). The scaling behavior of ONETEP was initially tested separately, since majority of the computational cost from the ONTETEP-i-PI interface resides in the electronic structure calculation. The overall ONETEP-i-PI interface was then tested in order to assess its overall performance.

Different Message-Passing-Interface(MPI)-OpenMultiprocessing(OpenMP) configurations were considered including simultaneous multi-threading(SMT). Due to the complexity of KNL being a self-hosted chip with more cores per CPU, various level of compiler optimization were tested including intrinsic vectorization capabilities. This approached was used in order to understand the benefits that KNL nodes give over conventional CPU such as example Broadwell, IvyBridge or Sandybridge processors.

All the results reported in this section refer to the average time necessary to complete one NGWFs + density kernel optimization in ONETEP. In the case of

the whole interface testing, the times reported are the one measured to perform one NGWFs + density kernel optimization in ONETEP, plus the communication through the socket, plus the MD step integration by i-PI.

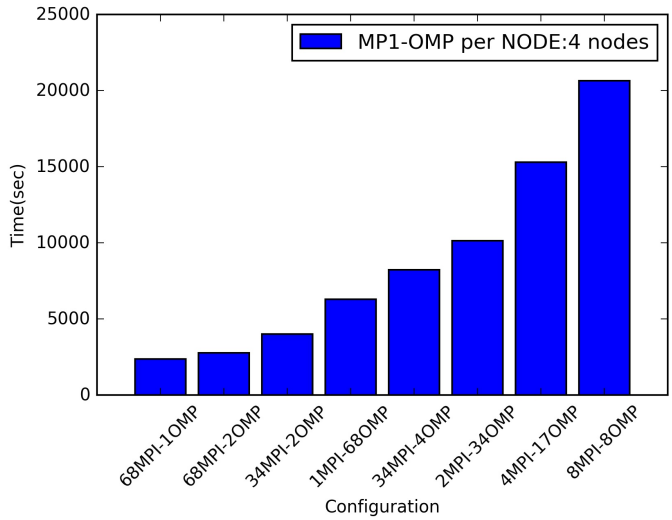


Figure 4.7: ONETEP timings for a single NGWFs + DKN optimization step running on 2 KNL nodes

The first test performed targeted the least-fitting unit that permitted to run a calculation to avoid a memory crush. Figure 4.7 reports the timing of those calculations. The configuration of 1-thread per MPI is the one that performs the best. Nonetheless the increment in time obtained using SMT (i.e. 2OMP per MPI process) is extremely small. This graphs reveals useful informations on how to run ONETEP and our interface when limited computational power is available. In fact exploiting the pure MPI set-up was possible to converge a $\approx$ 2000 atoms system using only 136 cores. While this graph provides a glimpse on the performance of ONETEP when the calculations are memory bound such trend is not be preserved when more nodes are employed.

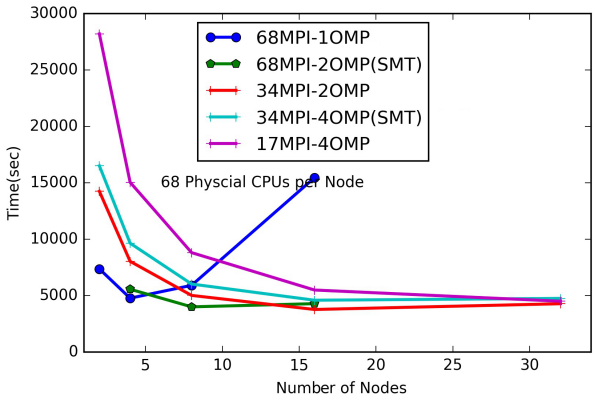


Figure 4.8: Strong scaling of the hybrid MPI-OpenMP ONETEP code for the calculation run on 2, 4, 8, 16 and 32 KNL nodes

Figure 4.8 shows the results for the strong scaling tests run on ONETEP only. These calculations were run employing $-O3$ compiler optimization. The Multi-Channel DRAM (i.e.MCDRAM) memory[2] was configured as cache memory to use on top of a RAM of 86 GB. The different hybrid MPI-OMP set-ups tested were: 68MPI/1OMP per node, 68MPI/2OMP per node making use of SMT, 34MPI/2OMP per node, 34MPI/4OMP per node with SMT and 17MPI/4OMP per node. The SMT here envisioned the use of multiple threads per core (specifically in our cases 2 OMP per core); as example in case of 2 nodes the configuration 68/2OMP resulted in overall 136 processes per 68 cores[3]. This range of configurations should help evaluate how the dynamic partitioning of resources and pipelines regulated by threads selectors affects the performances of ONETEP.

As already highlighted by Figure 4.7, the pure MPI set up (i.e. 68MPI/1OMP) leads to better timings in a regime where the overall memory (number of cores) is low if 4 or less KNL nodes are used this configuration should be applied. As soon as the simulations are run on a bigger scale, (probably they are not memory bound anymore) the set up 34MPI/2OMP becomes the best one. Given these results, this informed on the best MPI-OM configuration to be used to test the overall ONETEP-i-PI interface scaling. In addition from Figure 4.8, it can be seen that the SMT-enabled set up with 68 MPI processes per node (i.e. 68MPI/2OMP) behaves well in simulations using up to 8 nodes but degrades in efficiency in bigger calculations. On the contrary, the 34MPI/4OMP set-up appears to be as well behaved as its non-STM counterpart,(i.e. 34MPI-2OMP) with only a small increment in its

[2]MCDRAM could be configured as ”flat” other than ”cache”. This allows it to be utilized as part of the SDRAM.

[3]Each core can take up to 4 threads when using SMT. This results in an upper limit of 272 process for 68 cores.

performances. Last the 17MPI/4OMP configuration behaves poorly for a small number of cores but the trend depicted in Figure 4.8 shows a very favorable strong scaling leading to quasi-optimal performances for the calculation run on 32 nodes (only 34MPI-2OMP perform slightly better). However looking at its trend this configuration should be probably employed for calculations comprising more than 32 nodes. The dimension of our system of choice however did not permit this test since on 64 nodes we would have used more cores that the number of atoms. Such configuration a set up that is know to lead to a heavy reduction in performances.

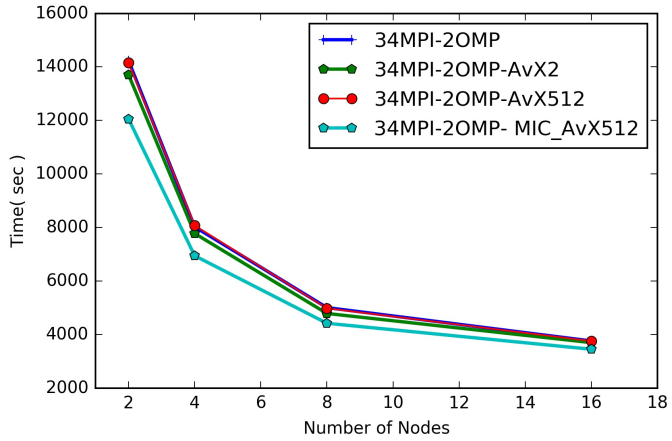


Figure 4.9: Strong scaling of the hybrid MPI-OpenMP ONETEP code using AVX2 and AVX512 vectorization flags for the best performing set up.

Figure 4.9 presents the comparative tests that were run, to monitor the ONETEP performances dependence with respect to the AVX2, AVX512 and MIC-AVX512 instruction sets. The most efficient hybrid MPI-OMP configuration (i.e. 34MPI/2OMP) was used through all these tests. The AVX2 instruction set expands integer commands to 256-bit and the float commands to 128-bit Single Instruction Multi Data (SIMD) registers. Similarly the AVX512 instruction set expands integer commands to 512-bit and the float commands to 256-bit SIMD registers. On top of these extensions KNL nodes allow for an additional set of instructions specific to their the Many Integrated Cores (MIC) architecture. The MIC set expands the vectorization instructions to the highly parallel processors comprising the KNL nodes. These processors also support 512-bit vectors, but with a new instruction set called IntelÂő Advanced Vector Extensions 512 (IntelÂő AVX-512).

Figure 4.9 shows a negligible increment in performance for the AVX2, AV512 vectorization; however an increment in performance of $\approx 10\%$ is observed

when the MIC instructions set is also considered. While this test only relies in the automatic vectorization of the ONETEP code it highlights the advantages of using the KNL processor. Further improvement could be obtained re-coding some of the main iterative cycles in the ONETEP source code. unfortunately such procedure is beyond the scope of this thesis but it could become the main focus of a future project.

During our project we also assessed the weak scaling and Linear scaling behavior of ONETEP. The weak scaling capabilities of ONETEP were examined using as test systems incrementally larger waters slabs. The different slabs studied were composed respectively by 1503, 2306, 3459, 4612 and 5765 atoms. The same parameters used for the strong scaling tests were adopted in these calculations. The same $\frac{Numberofatoms}{Numberofcores}$ ratio was kept for each point of the graph shown in Figure 4.9. The weak scaling trend was studied considering the two hybrid MPI-OMP set-ups that performed the best for a large number of cores (i.e. 34MPI/2OMP and 17MPI/4OMP).

Figure 4.10 shows weak scaling trends for both the set-ups considered. These results are far from perfect even taking into account the small calculation sample that has been probed. Nonetheless, the rate of increment in time-to-solution appears to slow down every time that the system gets bigger. Overall the 17MPI/4OMP set-up leads to better performances for bigger system/number of cores confirming the hypothesis deduced from Figure 4.8. This result has to be kept in mind in order to choose the optimal MPI-OMP configuration depending by the dimension of the system/calculation performed.

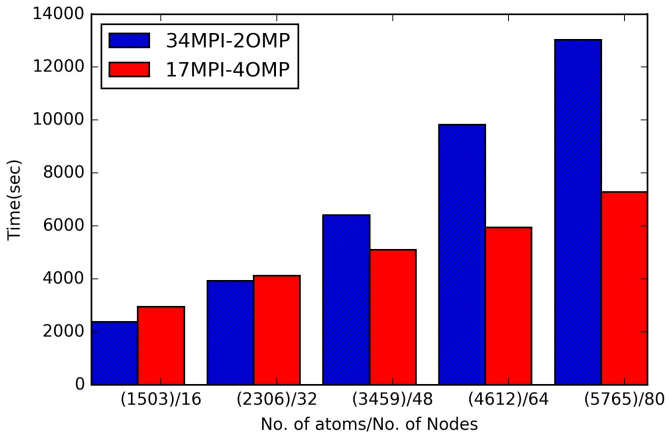


Figure 4.10: Weak scaling of the hybrid MPI-OpenMP ONETEP code considering the 32MPI/2OMP and the 17MPI/4OMP set-ups

ONETEP linear scaling was also studied using the same systems prepared for the

weak scaling analysis. The difference with respect to the weak scaling test resides in the fact that for linear scaling the number of atom is increased from a calculation to the next while the computational power is kept fixed. Hence the (number of atoms)/(Number of cores) ratio grows with the system size.

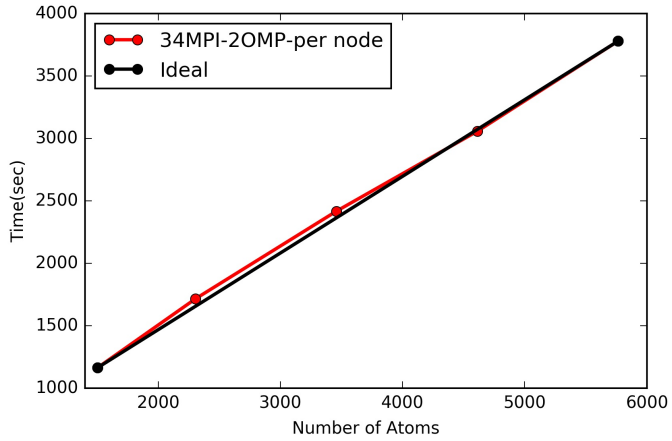


Figure 4.11: Linear scaling of the hybrid MPI-OpenMP ONETEP code considering the 32MPI/2OMP set-up

Figure 4.11 reports the results for the linear scaling test considering only the best performing MPI-OMP set-up (i.e. 34MPI/2OMP). The curve obtained is almost super imposed with the ideal one highlighting the scaling capabilities of ONETEP and its strength in treating incrementally bigger systems.

At last we studied the overall performance of the computational platform developed through all this thesis. During these calculations the parameters used for i-PI were the same used for the preliminary results calculations (see Table 4.1)

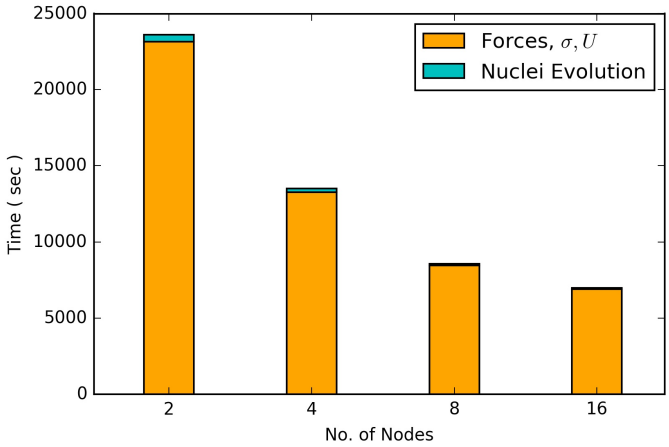


Figure 4.12: Strong scaling of the ONETEP-i-PI interface, for 1 MD step, considering the best MPI-OMP set up (i.e.34MPI/2OMP) for ONETEP. The overall timing contributions are divided between the forces, virial (i.e. σ) and potential energy (i.e. U) calculation in ONETEP and the Nuclei position evolution in i-PI. The socket communication time has been grouped with the i-PI timing for clarity.

Figure 4.12 shows the strong scaling behavior of the ONETEP-i-PI interface. The overall trend in timing mirrors exactly those obtained with ONETEP-only calculations. It is easy to observe how the time increment due to socket communication plus nuclei evolution is minimal, and decreases with the growth of the computational power used. This trend supports the idea of an "embarrassingly" parallel implementation of the *ab-inito*-PIMD with a rate limiting step determined by the forces calculation of the client code (i.e. ONETEP). Therefore the overall ONETEP-i-PI interface should scale linearly with respect to the system size. While many details and imprecisions are still present and have to be ironed-out (many errors still appears when the dynamics are propagated longer than few steps; this is still work in progress) these results potentially open the doors to peta-scale simulations of *ab-inito*-PIMD.

Chapter 5

Conclusion

In this thesis, an interface between the two computational packages i-PI and ONETEP was implemented in order to enable large scale simulations of *ab-initio* Path Integral Molecular Dynamics (ai-PIMD).

The *driver_ipi* subroutine was inserted in the ONETEP code allowing the communication with i-PI making by means of the socket protocol. *Driver_ipi* establishes a communication cycle between the two programs. This cycle permits i-PI to run like a server whiles ONETEP acts as a client. i-PI accounts for the numerical integration and the evolution of the nuclear degrees of freedom according to the equation of motion while ONETEP calculates and sends to i-PI the required quantities that are mainly forces, potential energy and the virial tensor.

Once the *driver_ipi* subroutine was completed the socket communication was thoroughly tested on different HPC machines. Several calculations on model systems were then run in order to verify the physical soundness of our hybrid (ONETEP/i-PI) simulations. To follow up, the ONETEP code takes on most of the workload of the *ab-initio* -PIMD. The interface was ported and tested on the A2 partition of CINECA's Tier-0 machine, Marconi. Strong, linear and weak scaling benchmarks were analyzed considering various hybrid MPI-OMP set-ups to find the most efficient configuration,(i.e. 34 MPI process per node, 2 OMP threads per MPI process), first, with ONETEP alone, and then with i-PI-ONETEP.

The results obtained for a system comprising more than 2000 atoms looks extremely promising. The strong scaling behavior of the i-PI-ONETEP interface is satisfactory and the increment in calculation time due to socket communication and the evolution of the nuclei positions is minimal and decreases with the magnitude of the computational power used. This interface implementation in fact takes full advantage of the "embarrassingly" parallel nature of the PIMD. In the current configuration then, the rate determining step of an *ab-initio*-PIMD calculation is defined by the forces calculation (i.e. self consistent cycle minimization of the density) in ONETEP. In principle while there is still a lot of work to do the

reported results confirm the possibility to expand *ab-initio*-PIMD to the near-petascale regime.

In conclusion, the tool developed through this interfacing procedure exploits the strengths of both the codes involved: the flexibility of i-PI as an interface to communicate with external programs and the linear-scaling-HPC-prone structure of ONETEP. We believe that the results reported here are an important complement to the ongoing effort to enable the petascale, exascale transition of different Density Functional Theory and Molecular Dynamic packages.

5.1 Further developments and future work

While the reported results are promising a lot of problems, and details still need to be tackled before an extended production run can be performed. As an example we are experiencing various errors due to memory crush when the *ab-initio*-PIMD is propagated for more that few steps. Frequent errors are also due to the Periodic Boundary Condition handling in i-PI that sometimes clash with ONETEP. The reasons behind both of these problems have still to be found. Our main focus in the next few moths will be on solving these issues in order to perform a full *ab-initio*-PIMD simulations on a system comprised by thousands of atoms. To the best of our knowledge such an achievement will have to be published in literature.

From a general point of view (and time/effort permitting) the interface between these two code should be tested and further expanded in order to exploit the full set of techniques that i-PI makes available. In addition to that as previously suggested further improvement in the computational performances could be obtained re-coding some of the main iterative cycles in the ONETEP source code in order to take advantage of the full vectorization capability of the KNL nodes.

Appendix A

Appendix

A.1 Numerical Stability

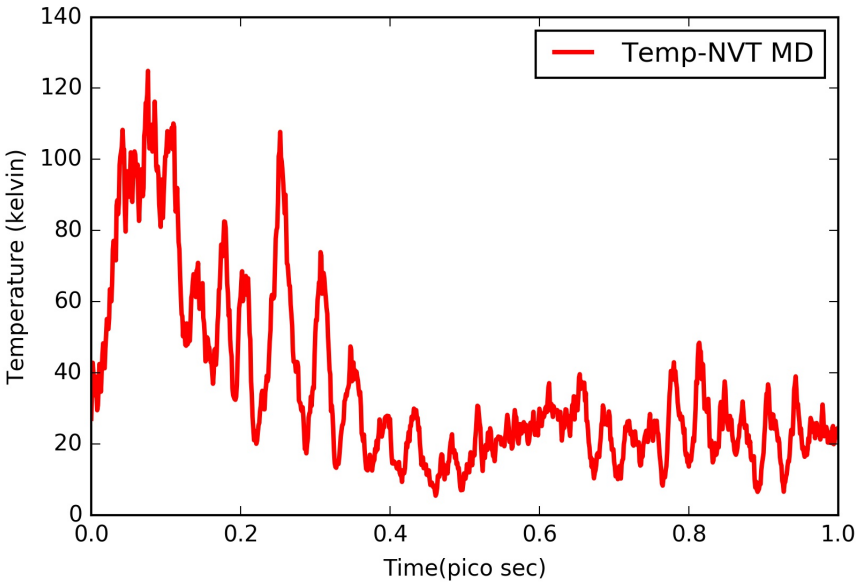


Figure A.1: Classical MD: Temperature

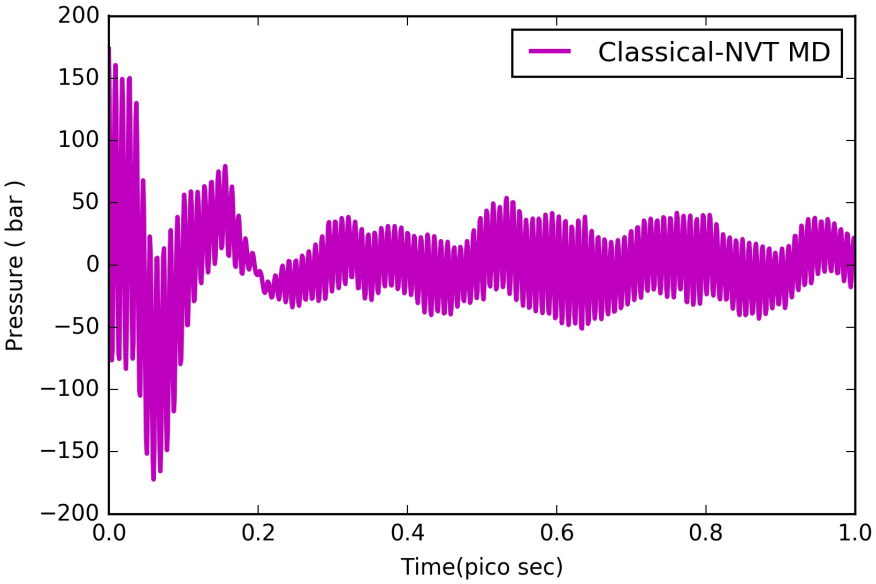


Figure A.2: Classical MD: Pressure

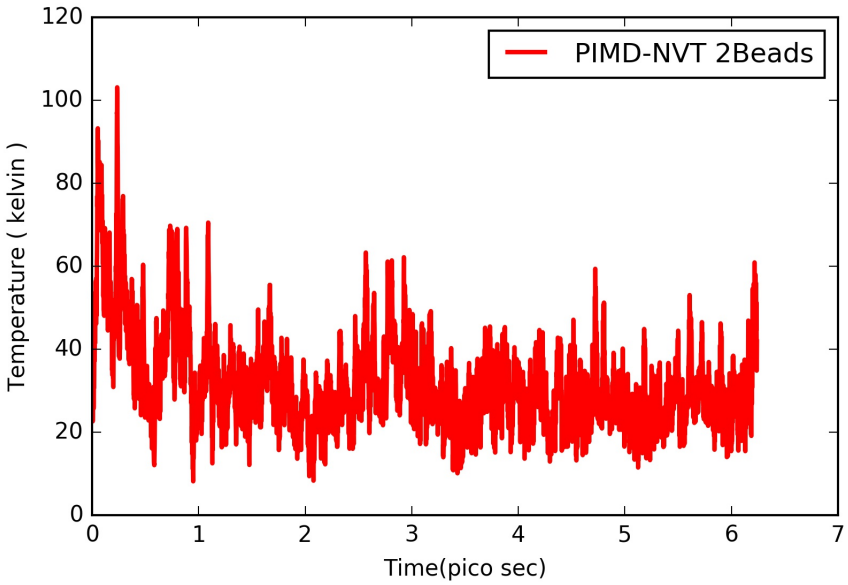


Figure A.3: PIMD(2 beads): Temperature

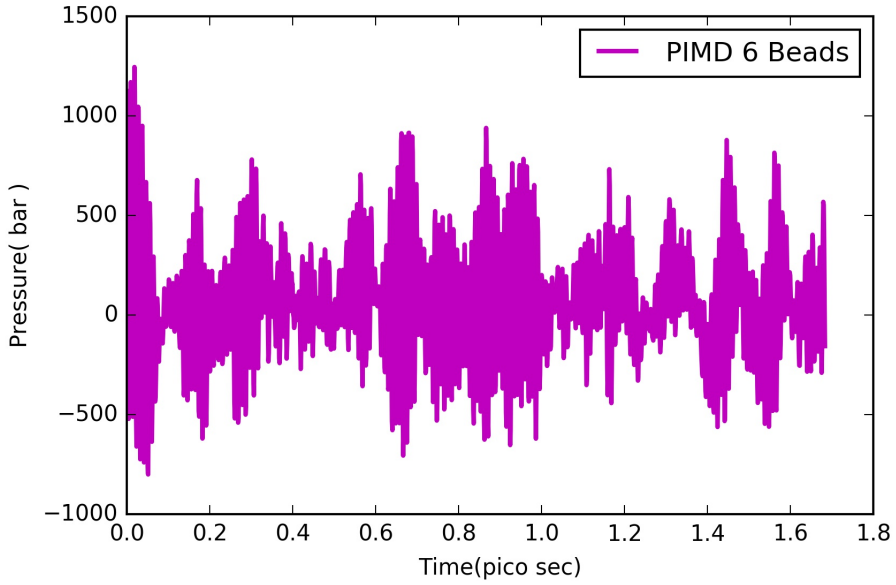


Figure A.4: PIMD(2 beads): Pressure

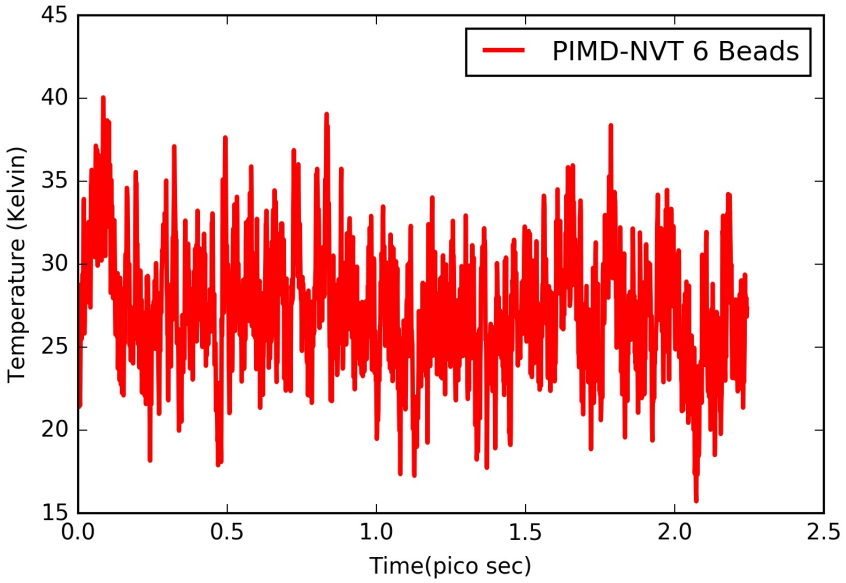


Figure A.5: PIMD(6 beads): Temperature

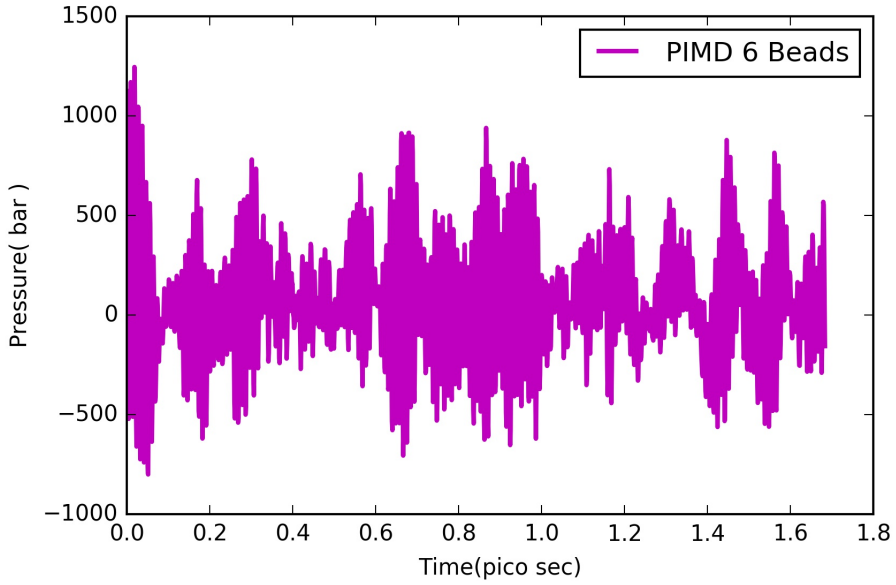


Figure A.6: PIMD(6 beads): Pressure

A.2 Scalability

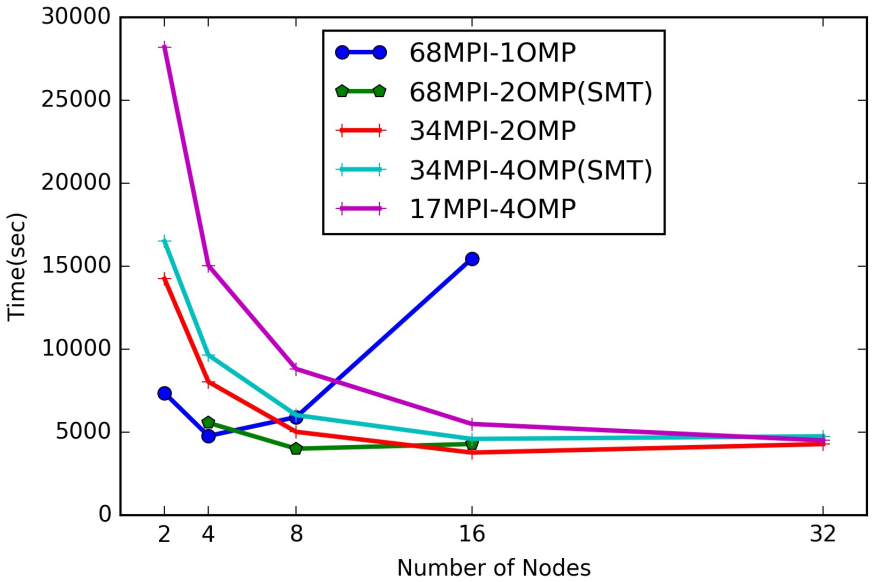


Figure A.7: Strong scaling with SMT

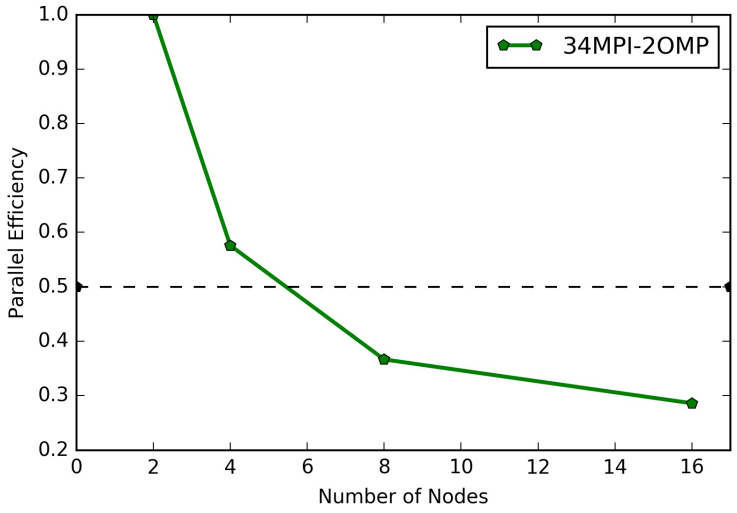


Figure A.8: Parallel Efficiency on 2306-atom air-water(O + H) interface on KNL partition on MARCONI using 2-OMP threads per MPI process

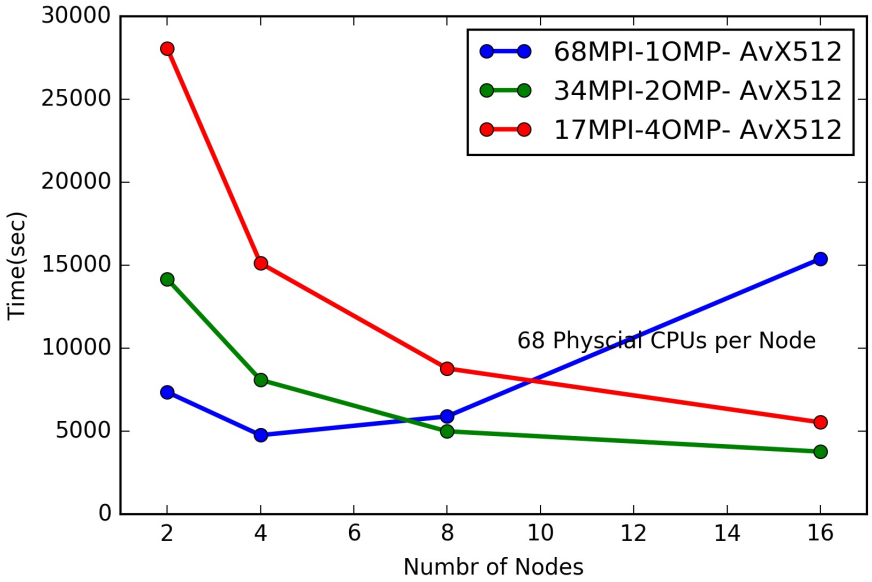


Figure A.9: MPI-OMP configuration on 2306 atoms system running 4 KNL nodes

Bibliography

[1] A. Hospital, J. R. Goñi, M. Orozco, and J. L. Gelpí, "Molecular dynamics simulations: advances and applications," *Advances and applications in bioinformatics and chemistry: AABC*, vol. 8, p. 37, 2015.

[2] C. C. Fischer, K. J. Tibbetts, D. Morgan, and G. Ceder, "Predicting crystal structure by merging data mining with quantum mechanics," *Nature materials*, vol. 5, no. 8, pp. 641–646, 2006.

[3] M. Ceriotti, J. More, and D. E. Manolopoulos, "i-pi: A python interface for ab initio path integral molecular dynamics simulations," *Computer Physics Communications*, vol. 185, no. 3, pp. 1019–1026, 2014.

[4] R. Car and M. Parrinello, "Unified approach for molecular dynamics and density-functional theory," *Physical review letters*, vol. 55, no. 22, p. 2471, 1985.

[5] G. Kresse and J. Hafner, "Ab initio molecular dynamics for liquid metals," *Physical Review B*, vol. 47, no. 1, p. 558, 1993.

[6] G. Kresse and J. Hafner, "Ab initio molecular-dynamics simulation of the liquid-metal–amorphous-semiconductor transition in germanium," *Physical Review B*, vol. 49, no. 20, p. 14251, 1994.

[7] J. A. Morrone and R. Car, "Nuclear quantum effects in water," *Physical review letters*, vol. 101, no. 1, p. 017801, 2008.

[8] E. A. Long and J. Kemp, "The entropy of deuterium oxide and the third law of thermodynamics. heat capacity of deuterium oxide from 15 to 298 k. the melting point and heat of fusion," *Journal of the American Chemical Society*, vol. 58, no. 10, pp. 1829–1834, 1936.

[9] C.-K. Skylaris, P. D. Haynes, A. A. Mostofi, and M. C. Payne, "Introducing onetep: Linear-scaling density functional simulations on parallel computers," *The Journal of chemical physics*, vol. 122, no. 8, p. 084119, 2005.

[10] M. Ceriotti and D. E. Manolopoulos, "Efficient first-principles calculation of the quantum kinetic energy and momentum distribution of nuclei," *Physical review letters*, vol. 109, no. 10, p. 100604, 2012.

[11] V. Kapil, J. VandeVondele, and M. Ceriotti, "Accurate molecular dynamics and nuclear quantum effects at low cost by multiple steps in real and imaginary time: Using density functional theory to accelerate wavefunction methods," *The Journal of chemical physics*, vol. 144, no. 5, p. 054111, 2016.

[12] M. Rossi, P. Gasparotto, and M. Ceriotti, "Anharmonic and quantum fluctuations in molecular crystals: A first-principles study of the stability of paracetamol," *Physical Review Letters*, vol. 117, sep 2016.

[13] B. Cheng and M. Ceriotti, "Direct path integral estimators for isotope fractionation ratios," *The Journal of Chemical Physics*, vol. 141, p. 244112, dec 2014.

[14] M. Ceriotti, G. Bussi, and M. Parrinello, "Colored-noise thermostats à la carte," *Journal of Chemical Theory and Computation*, vol. 6, pp. 1170–1180, apr 2010.

[15] Y. Luo, A. Zen, and S. Sorella, "Ab initio molecular dynamics with noisy forces: Validating the quantum monte carlo approach with benchmark calculations of molecular vibrational properties," *The Journal of Chemical Physics*, vol. 141, p. 194112, nov 2014.

[16] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," *Phys. Rev.*, vol. 136, pp. B864–B871, Nov 1964.

[17] W. Kohn and L. J. Sham, "Self-consistent equations including exchange and correlation effects," *Phys. Rev.*, vol. 140, pp. A1133–A1138, Nov 1965.

[18] R. S., "Citation statistics from 110 years of physical review," *Phys. Today*, vol. 58, p. 49, Nov 2005.

[19] J. Junquera and P. Ghosez, "Critical thickness for ferroelectricity in perovskite ultrathin films," *Nature*, vol. 422, p. 506, Apr 2003.

[20] M. T. Green, J. H. Dawson, and H. B. Gray, "Oxoiron(iv) in chloroperoxidase compound ii is basic: Implications for p450 chemistry," *Science*, vol. 304, no. 5677, pp. 1653–1656, 2004.

[21] M. J. Alf, D.and Gillan and G. D. Price, "The melting curve of iron at the pressures of the earth's core from ab initio calculations," *Nature*, vol. 401, p. 462, 1999.

[22] D. R. Bowler, T. Miyazaki, and M. J. Gillan, "Recent progress in linear scaling ab initio electronic structure techniques," *Journal of Physics: Condensed Matter*, vol. 14, no. 11, p. 2781, 2002.

[23] S. Goedecker, "Linear scaling electronic structure methods," *Rev. Mod. Phys.*, vol. 71, pp. 1085–1123, Jul 1999.

[24] G. Galli, "Linear scaling methods for electronic structure calculations and quantum molecular dynamics simulations," *Current Opinion in Solid State and Materials Science*, vol. 1, no. 6, pp. 864 – 874, 1996.

[25] M. C. Payne, M. P. Teter, D. C. Allan, T. A. Arias, and J. D. Joannopoulos, "Iterative minimization techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients," *Rev. Mod. Phys.*, vol. 64, pp. 1045–1097, Oct 1992.

[26] E. Prodan and W. Kohn, "Nearsightedness of electronic matter," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 33, pp. 11635–11638, 2005.

[27] W. Kohn, "Density functional and density matrix method scaling linearly with the number of atoms," *Phys. Rev. Lett.*, vol. 76, pp. 3168–3171, Apr 1996.

[28] W. Kohn, "Analytic properties of bloch waves and wannier functions," *Phys. Rev.*, vol. 115, pp. 809–821, Aug 1959.

[29] J. D. Cloizeaux, "Analytical properties of n-dimensional energy bands and wannier functions," *Phys. Rev.*, vol. 135, pp. A698–A707, Aug 1964.

[30] S. Ismail-Beigi and T. A. Arias, "Locality of the density matrix in metals, semiconductors, and insulators," *Phys. Rev. Lett.*, vol. 82, pp. 2127–2130, Mar 1999.

[31] C.-K. Skylaris, A. A. Mostofi, P. D. Haynes, O. Diéguez, and M. C. Payne, "Nonorthogonal generalized wannier function pseudopotential plane-wave method," *Phys. Rev. B*, vol. 66, p. 035119, Jul 2002.

[32] R. McWeeny, "Some recent advances in density matrix theory," *Rev. Mod. Phys.*, vol. 32, pp. 335–369, Apr 1960.

[33] C. Fonseca Guerra, J. G. Snijders, G. te Velde, and E. J. Baerends, "Towards an order-n dft method," *Theoretical Chemistry Accounts*, vol. 99, pp. 391–403, Nov 1998.

[34] G. E. Scuseria, "Linear scaling density functional calculations with gaussian orbitals," *The Journal of Physical Chemistry A*, vol. 103, no. 25, pp. 4782–4790, 1999.

[35] M. Challacombe, "A simplified density matrix minimization for linear scaling self-consistent field theory," *The Journal of Chemical Physics*, vol. 110, no. 5, pp. 2332–2342, 1999.

[36] E. Hernández, M. J. Gillan, and C. M. Goringe, "Basis functions for linear-scaling first-principles calculations," *Phys. Rev. B*, vol. 55, pp. 13485–13493, May 1997.

[37] C.-K. Skylaris, A. A. Mostofi, P. D. Haynes, O. Diéguez, and M. C. Payne, "Nonorthogonal generalized wannier function pseudopotential plane-wave method," *Phys. Rev. B*, vol. 66, p. 035119, Jul 2002.

[38] N. Marzari, D. Vanderbilt, and M. C. Payne, "Ensemble density-functional theory for ab initio molecular dynamics of metals and finite-temperature insulators," *Phys. Rev. Lett.*, vol. 79, pp. 1337–1340, Aug 1997.

[39] P. D. Haynes and M. C. Payne, "Corrected penalty-functional method for linear-scaling calculations within density-functional theory," *Phys. Rev. B*, vol. 59, pp. 12173–12176, May 1999.

[40] X.-P. Li, R. W. Nunes, and D. Vanderbilt, "Density-matrix electronic-structure method with linear system-size scaling," *Phys. Rev. B*, vol. 47, pp. 10891–10894, Apr 1993.

[41] P. D. Haynes, C.-K. Skylaris, A. A. Mostofi, and M. C. Payne, "Density kernel optimization in the onetep code," *Journal of Physics: Condensed Matter*, vol. 20, no. 29, p. 294207, 2008.

[42] N. Marzari, A. A. Mostofi, J. R. Yates, I. Souza, and D. Vanderbilt, "Maximally localized wannier functions: Theory and applications," *Rev. Mod. Phys.*, vol. 84, pp. 1419–1475, Oct 2012.

[43] A. A. Mostofi, P. D. Haynes, C.-K. Skylaris, and M. C. Payne, "Preconditioned iterative minimization for linear-scaling electronic structure calculations," *The Journal of Chemical Physics*, vol. 119, no. 17, pp. 8842–8848, 2003.

[44] C.-K. Skylaris, A. A. Mostofi, P. D. Haynes, C. J. Pickard, and M. C. Payne, "Accurate kinetic energy evaluation in electronic structure calculations with localized functions on real space grids," *Computer Physics Communications*, vol. 140, no. 3, pp. 315 – 322, 2001.

[45] C.-K. Skylaris, P. D. Haynes, A. A. Mostofi, and M. C. Payne, "Implementation of linear-scaling plane wave density functional theory on parallel computers," *physica status solidi (b)*, vol. 243, no. 5, pp. 973–988, 2006.

[46] A. A. Mostofi, C.-K. Skylaris, P. D. Haynes, and M. C. Payne, "Total-energy calculations on a real space grid with localized functions and a plane-wave basis," *Computer Physics Communications*, vol. 147, no. 3, pp. 788 – 802, 2002.

[47] R. P. Feynman and A. Hibbs, *Quantum mechanics and path integrals [by] RP Feynman [and] AR Hibbs*. McGraw-Hill, 1965.

[48] B. J. Berne and D. Thirumalai, "On the simulation of quantum systems: path integral methods," *Annual Review of Physical Chemistry*, vol. 37, no. 1, pp. 401–424, 1986.

[49] M. E. Tuckerman, B. J. Berne, G. J. Martyna, and M. L. Klein, "Efficient molecular dynamics and hybrid monte carlo algorithms for path integrals," *The Journal of Chemical Physics*, vol. 99, no. 4, pp. 2796–2808, 1993.

[50] H. Trotter, "An elementary proof of the central limit theorem," *Archiv der Mathematik*, vol. 10, no. 1, pp. 226–234, 1959.

[51] L. S. Schulman, *Techniques and applications of path integration*. Courier Corporation, 2012.

[52] C. Burnham, G. Reiter, J. Mayers, T. Abdul-Redah, H. Reichert, and H. Dosch, "On the origin of the redshift of the oh stretch in ice ih: evidence from the momentum distribution of the protons and the infrared spectral density," *Physical Chemistry Chemical Physics*, vol. 8, no. 34, pp. 3966–3977, 2006.

[53] C. Burnham, D. Anick, P. Mankoo, and G. Reiter, "The vibrational proton potential in bulk liquid water and ice," *The Journal of chemical physics*, vol. 128, no. 15, p. 154519, 2008.

[54] M. Ceriotti, D. E. Manolopoulos, and M. Parrinello, "Accelerating the convergence of path integral dynamics with a generalized langevin equation," *The Journal of Chemical Physics*, vol. 134, p. 084104, feb 2011.

[55] M. Ceriotti and T. E. Markland, "Efficient methods and practical guidelines for simulating isotope effects," *The Journal of chemical physics*, vol. 138, no. 1, p. 014112, 2013.

[56] G. Van Rossum *et al.*, "Python programming language.," in *USENIX Annual Technical Conference*, vol. 41, p. 36, 2007.

[57] X. Gonze, R. Stumpf, and M. Scheffler, "Analysis of separable potentials," *Phys. Rev. B*, vol. 44, pp. 8503–8513, Oct 1991.