



The VENUS/NWChem software package. Tight coupling between chemical dynamics simulations and electronic structure theory[☆]



Upakarasamy Lourderaj^a, Rui Sun^b, Swapnil C. Kohale^b, George L. Barnes^c,
Wibe A. de Jong^d, Theresa L. Windus^e, William L. Hase^{b,*}

^a School of Chemical Sciences, National Institute of Science Education and Research, Bhubaneswar 751005, India

^b Department of Chemistry and Biochemistry, Texas Tech University, Lubbock, TX 79409, United States

^c Department of Chemistry and Biochemistry, Siena College, Loudonville, NY 12211, United States

^d Environmental Molecular Science Laboratory, Pacific Northwest National Laboratory, Richland, WA 99352, United States

^e Department of Chemistry, Iowa State University, Ames, IA 50011, United States

ARTICLE INFO

Article history:

Received 9 September 2013

Received in revised form

14 November 2013

Accepted 18 November 2013

Available online 26 November 2013

Keywords:

Direct dynamics

Classical trajectories

Molecular simulation

ABSTRACT

The interface for VENUS and NWChem, and the resulting software package for direct dynamics simulations are described. The coupling of the two codes is considered to be a tight coupling since the two codes are compiled and linked together and act as one executable with data being passed between the two codes through routine calls. The advantages of this type of coupling are discussed. The interface has been designed to have as little interference as possible with the core codes of both VENUS and NWChem. VENUS is the code that propagates the direct dynamics trajectories and, therefore, is the program that drives the overall execution of VENUS/NWChem. VENUS has remained an essentially sequential code, which uses the highly parallel structure of NWChem. Subroutines of the interface that accomplish the data transmission and communication between the two computer programs are described. Recent examples of the use of VENUS/NWChem for direct dynamics simulations are summarized.

Program summary

Program title: VENUS/NWChem

Catalogue identifier: AERS_v1_0

Program summary URL: http://cpc.cs.qub.ac.uk/summaries/AERS_v1_0.html

Program obtainable from: CPC Program Library, Queen's University, Belfast, N. Ireland

Licensing provisions: Open Source Educational Community License

No. of lines in distributed program, including test data, etc.: 10,831,970

No. of bytes in distributed program, including test data, etc.: 77,141,871

Distribution format: tar.gz

Programming language: Fortran 77 with some C in NWChem, MPI.

Computer: All Linux based workstations and parallel supercomputers.

Operating system: Linux.

Has the code been vectorized or parallelized?: Venus is a sequential code; NWChem can run in parallel.

Classification: 16.8.

Subprograms used:

Cat Id	Title	Reference
AEGL_v1_0	NWChem	CPC 181(2010)1477

[☆] This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (<http://www.sciencedirect.com/science/journal/00104655>).

* Correspondence to: Box 41061, Lubbock, TX 79409-1061, United States. Tel.: +1 806 742 3152; fax: +1 806 742 1240.

E-mail address: bill.hase@ttu.edu (W.L. Hase).

Nature of problem:

Direct dynamics simulations play an important role in investigating and understanding atomic-level chemical dynamics information such as atomistic reaction mechanisms, unimolecular and bimolecular rate constants, intramolecular vibrational energy redistribution rates, etc. The ability to couple direct dynamics with electronic structure methods brings a level of fidelity to the simulations that is important for complex systems. However, a tight coupling between two codes that have their own development teams and schedules can be challenging.

Solution method:

The VENUS/NWChem interface is designed to link the general electronic structure program (NWChem) and classical chemical dynamics simulation program (VENUS) to perform direct dynamics simulation in which the trajectories “on the fly” with the potential and its derivatives obtained directly from electronic structure theory. One of the design goals is to build interfaces that require as little interference in NWChem and VENUS as possible so that each of the code developments can continue independently. This is especially important since VENUS is currently a sequential code and NWChem is a parallel code and being able to compute the energies, gradients, and Hessian in parallel is an important aspect of making the software useful to users. In this manuscript, the tight coupling interface between the two codes is described and examples of its use are given. In the classical chemical dynamics simulation an ensemble of trajectories is calculated, and the initial sampling represents the conditions of the reactants for the chemical reaction under investigation. Each trajectory is evaluated by numerically integrating either Hamilton's or Newton's equations of motion. The Schrödinger equation is solved and the energy and energy gradient are calculated in the electronic structure program (NWChem), and this information is passed to the classical trajectory program (VENUS) to solve the equations of motion.

Additional comments:

Full documentation is provided in the distribution file. This includes a README file giving the names and brief description of all the files that make up the package and instructions on the installation and execution of the program. Sample input and output data for test run will also be provided.

The software is free to download and use once a signed license agreement has been received. The agreement will be displayed when the program is requested.

Running time:

The running time depends on the size of the chemical system, simulation time, complexity of the ab initio method and number of CPUs. The ab initio method is the most time consuming part of each step in the calculations and scaling for different types of systems and levels of theory is available in Valiev et al. (2010). Again, there are many factors that affect the running time for the full simulation and it can range from several hours for simulations of a few atoms with DFT and a small basis set running on a single compute node to several days for the simulation of tens of heavy atoms with larger basis set running parallel.

© 2013 Elsevier B.V. All rights reserved.

1. Introduction

Classical trajectory chemical dynamics simulations provide an atomistic understanding of a variety of chemical processes including energy transfer and chemical reaction in gas-phase [1] and gas-surface [2] collisions, unimolecular decomposition [3] and conformational change [4], intramolecular vibrational energy distribution (IVR) [5], and condensed phase intermolecular and intramolecular energy transfer and chemical reaction [6,7]. An ensemble of trajectories is propagated in such a simulation, with each trajectory determined by numerically integrating the classical equations of motion [8–10]. What is required for this calculation is the potential energy function (or surface) for the chemical process, and the potential's gradient (and also the Hessian for some approaches) [11–13]. Analytic molecular mechanical (MM) potential energy functions are often used to simulate the dynamics of chemical processes which do not involve bond rupture [14]. Specific analytic potential energy functions are needed to represent bimolecular and unimolecular reactions, and developing such a function which accurately represents a chemical process with multiple pathways, transition states, and minima is a formidable task [15,16].

The direct dynamics approach for performing a classical trajectory simulation was first implemented, using the CNDO semiempirical method, to study the ${}^1\text{CH}_2 + \text{H}_2 \rightarrow \text{CH}_4$ reaction [17]. The initial ab initio direct dynamics trajectory simulation was for the $\text{H}^- + \text{CH}_4 \rightarrow \text{CH}_4 + \text{H}^- \text{S}_{\text{N}}2$ reactions [18].

Electronic structure calculations have also been used directly to obtain the potential energy surface information for following minimum energy paths and calculating transition state theory rate constant [19–22]. A general and broadly applicable approach for performing a classical trajectory chemical dynamics simulation is to obtain the potential energy function, and its gradient and Hessian, directly from an electronic structure theory [10]. The latter is a component of a quantum chemistry software package such as NWChem [23], GAMESS [24], or MOLPRO [25], so that the software needed for such a chemical dynamics simulation requires one to interface the technology of a chemical dynamics simulation with that of quantum chemistry. The resulting simulation is referred to as “direct dynamics” [10], since the potential energy, gradient, and Hessian are obtained directly without the need for an analytic potential energy function. If the complete potential energy function for the chemical process is represented by an electronic structure theory, the simulation is referred to as quantum mechanics (QM) direct dynamics. For some chemical processes the potential energy function may be represented by a combination of a QM model and MM analytic potential energy terms [10].

This paper describes the interface of two codes – VENUS [26,27] for the propagation of the trajectories and NWChem [23] for the QM evaluation of energies, gradients and Hessians – to enable QM direct dynamics. The coupling of the two codes is considered to be a tight coupling, since the two codes are compiled and linked together and act as one executable with data being passed between

the two codes through routine calls. This is in contrast to a loose coupling where the two codes would work relatively independent of one another. In a loose coupling, a script or shell call would direct the execution of the codes and the data would be passed from one code to the other through files. The tight code coupling has several advantages in this case. Both VENUS and NWChem have a general interface to external codes that can be exploited (although modifications were required as described below) and overall performance will be higher since files will not be generated and software will not have initialization startup costs each time it is needed.

2. General VENUS/electronic structure theory interface for direct dynamics simulations

The flowchart for calculating a classical trajectory with the VENUS computer program is given in Fig. 1. Execution of the program begins by setting up the classical trajectory calculation. VENUS can simulate the dynamics of unimolecular reactions, bimolecular collisions and reactions, gas–surface collisions and reactions, and dynamics initiated at a potential energy barrier, and there are several ways of choosing initial conditions for these simulations. The vibrational energy for a polyatomic molecule may be chosen by: (i) sampling its classical or quantum microcanonical ensemble; (ii) selecting a specific normal mode or local mode vibrational state; or (iii) sampling its normal mode Boltzmann distribution of energy. The rotational energy can be chosen by adding $RT/2$ of energy about each rotation axis or by sampling the Boltzmann rotational energy distribution. For a diatomic molecule the initial conditions are for a specific vibration/rotation state. For bimolecular collisions/reactions, the relative translational energy can be either fixed or chosen from its Boltzmann distribution. For gas–surface collisions/reactions, initial conditions for the projectile may be sampled using one of the methods described above, while the surface is thermally equilibrated to the desired surface temperature T_{surf} . In addition, a simulation may be performed with the trajectories initialized at a potential energy barrier, and the trajectories directed toward either reactants or products or randomly selected to move in both of these two directions. Initial conditions for trajectories at a potential energy barrier may be chosen from a quantum microcanonical or canonical ensemble, or by sampling the $(3N - 7)$ vibrational degrees of freedom at the barrier with one of the above methods and putting a fixed amount of energy in the reaction coordinate translation.

The selection of initial conditions involves calls to evaluate the potential energy, and its gradient and Hessian, to select the vibrational energy within the molecule for the chosen sampling scheme. Once the trajectory simulation is initiated, the numerical integration of the trajectory involves calls to obtain the gradient for a gradient-based integration algorithm such as Adams–Moulton [28] and velocity Verlet [29], and also the Hessian for a local quadratic Hessian-based integrator [11–13].

For a traditional trajectory simulation, the potential energy and its gradient and Hessian are evaluated using an analytic functional form for the potential energy surface (PES). However, for a direct dynamics simulation, the potential energy, gradient, and Hessian are represented by an electronic structure theory and obtained directly from an electronic structure computer program. Hence, for a direct dynamics simulation, information such as the theoretical method, basis set, atomic numbers, charge, and spin multiplicity of the electronic state must be identified for the electronic structure program. Once the necessary task is completed by the electronic structure program, the evaluated energy/gradient/Hessian are passed back to VENUS.

The transmission of information between VENUS and the electronic structure program is performed by a set of interface

routines. For example, in selecting initial conditions vibrational frequencies are required. VENUS passes the necessary information for the evaluation of the Hessian to the electronic structure program through the interface. The electronic structure program computes the Hessian matrix and passes it back to VENUS via the interface, and the frequencies are then calculated within VENUS. The trajectory is numerically integrated within VENUS. If a gradient-based integration algorithm is used, the gradient for the current position on the PES is evaluated by the electronic structure program and transmitted to VENUS. The potential energy is also transmitted so that it may be reported during the trajectory integration. For Hessian-based integration, for which a local quadratic PES is constructed about the current position on the PES, the potential energy, gradient, and Hessian are transmitted from the electronic structure theory program to VENUS for a given point on the PES.

A trajectory simulation which includes both an electronic structure theory quantum mechanical (QM) component and an analytic potential energy component may be performed with VENUS/NWChem. The analytic potential energy component often consists of MM functions which do not allow reaction; however, VENUS has analytic potentials which do allow reaction [16]. Such a direct dynamics simulation with both QM and analytic potential energy functions is often called QM/MM, regardless of the form of the analytic potential functions. The Hessian-based integrator is not applicable to a QM/MM simulation, which must be performed with a gradient-based integrator. The trajectory is integrated by obtaining both the QM and MM components to the gradient. If the QM and MM components of the simulation are truly separable so that the total potential energy may be written as $V = V_{\text{QM}} + V_{\text{MM}}$, the simulation is referred to as QM+MM [10]. For a QM/MM simulation there are couplings between the QM and MM components.

VENUS/NWChem performs adiabatic direct dynamics, with the chemical system remaining on the PES of a single electronic state. To assist in assuring this, the initial guess of the wave function for the electronic structure calculation is first obtained from the optimized wave function for the geometry in the previous trajectory integration step. Another advantage of this feature is to improve the efficiency of the self-consistent field (SCF) calculation, since the SCF starts from a guess that is very close to the optimized wavefunction. If the trajectory is in the vicinity of an avoided crossing between the PESs of different electronic states, the nature of the wave function for the trajectory's PES may change appreciably and the wave function for the previous trajectory step may not be a good initial guess for the current wave function. If this is the case, other approaches may be needed for the initial guess. The VENUS/NWChem interface is constructed to handle such situations by increasing the number of SCF iterations to obtain convergence, by decreasing the integration step size, or changing the initial guess, e.g. to the Hückel guess. However, if there is an error in the total energy due to a significant change in the nature of the electronic structure, a trajectory will stop.

3. General NWChem external interface

NWChem [23] is a high performance computational chemistry code that has extensive functionality in QM (both Gaussian and pseudopotential basis sets), MM and QM/MM. In addition, several statistical methods, such as molecular dynamics and Monte Carlo, have been incorporated into NWChem to enable difficult chemistry questions to be answered. However, a direct dynamics functionality has not been *directly* implemented into NWChem making the interface with VENUS a very valuable one. NWChem is based on a non-uniform memory access (NUMA) parallelization model that is provided by the Global Array (GA) [30] toolkit based

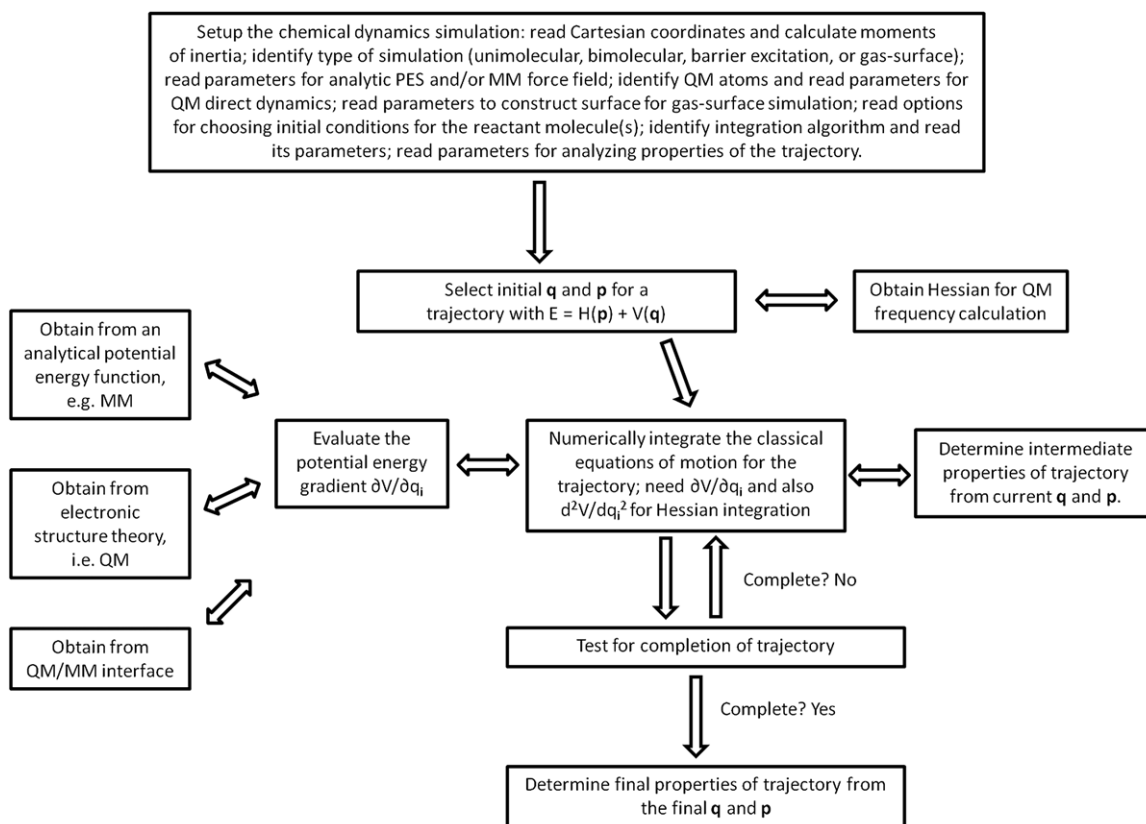


Fig. 1. Flowchart for the VENUS chemical dynamics computer program, for which an ensemble of classical trajectories is calculated. The potential energy function may be analytic (e.g. MM), quantum mechanical (QM), or a QM+MM or QM/MM combination.

on the aggregate remote memory copy interface (ARMCI) [31]. This model has allowed several of the QM methods to scale to petascale computations in an efficient manner.

In theory QM/MM direct dynamics with VENUS/NWChem could be performed with the functionality in NWChem. However, this would require integrating complex topology files for the NWChem input with the VENUS input and this has not been done. Thus, with the current VENUS/NWChem package, QM/MM simulations may only be performed with the VENUS functionality as described above, which does not include electrostatic embedding. This latter QM/MM potential is expected to be important for several simulation types and in future work it is of interest to integrate the QM/MM functionality in NWChem into VENUS/NWChem.

Since integration of software has become important to the computational chemistry community as a whole, a general interface to basic functionality has been built into NWChem. Initially, the targets of these interfaces were the coupling with VENUS, described in this paper, and the use of NWChem in the Common Component Architecture (CCA) project [32]. In the latter work, the interfaces were originally intended to enable multiple quantum chemistry codes – NWChem, MPQC [33], and GAMESS – to interoperate [34] and interface with general optimization codes such as TAO [35] in a dynamically linked computational environment. However, as the work progressed, these interfaces were also used to allow integration of quantum mechanics codes with the MM functionality within NWChem [36]. Most of the quantum mechanics interfaces required no changes to enable this capability, thus, showing the flexibility and functionality of the original design of the QM interface.

Since the general interfaces have been described elsewhere [34], only a brief description of the interface routines related to this work will be provided here. The first five routines include new

routines or modified routines due to the specific needs to expose a parallel environment to the VENUS code.

net_init: this subroutine initializes the parallel environment of NWChem and passes back the number of processes and the current process' identity. The latter is a number from 0 to the number of processes minus one and gives each individual process in the parallel environment a unique identity. This routine is a new routine that was not in the original CCA interface and is needed since VENUS does not have a parallel environment of its own. The parallel environment start up was originally part of *initialize_qm* and did not pass information about the parallel environment.

initialize_qm: this subroutine initializes the rest of the NWChem computational environment that includes the runtime database (rtdb), the memory to be used, and the details for the atomic system to be computed. This routine is a bit more general than its original version developed for the CCA to allow for more information to be passed. The types of information passed are the level of theory (for example, HF, SCF, DFT, MP2, etc.), SCF options (convergence threshold, iteration limit, etc.), initial molecular orbital guess (built-in guess as Hückel or pre-computed guess to assure the trajectory remains in a specific electronic state), basis set name, number of atoms, Cartesian coordinates, labels for the atoms, overall charge of the chemical system, the level of output to be printed to the output, the memory requirements (defaults are provided by NWChem if zeros are passed through the interface), and, optionally, the name of a file that can be read for additional NWChem input parameters that are not passed in through the calling arguments.

net_final: this subroutine is also new and stops the parallel environment. Again, this is due to the need for VENUS to keep the parallel environment available after the QM part has completed. This function was originally part of the *finalize_qm* function.

finalize_qm: this function with no arguments deallocates memory and closes the rtdb, effectively stopping the QM code from running unless a new *initialize_qm* is called. Usually this routine is only called once.

net_broadcast: this is a new subroutine that allows VENUS to use the GA functionality of NWChem to broadcast information within VENUS.

evaluateObjective: this function takes a set of Cartesian coordinates and returns an energy.

evaluateGradient: this function takes a set of Cartesian coordinates and returns the gradient.

evaluateObjectiveAndGradient: this function takes a set of Cartesian coordinates and returns the energy and the gradient.

evaluateHessian: this function takes a set of Cartesian coordinates and returns the Hessian.

4. The VENUS/NWCHEM interface

When designing the VENUS/NWChem interface, several design goals were developed to allow the design to be flexible and useful. These design goals are:

1. The interface needs to have as little interference as possible with the core code of both VENUS and NWChem.
2. VENUS is to remain an essentially sequential code that uses parallel QM codes. This requires that VENUS becomes parallel aware, but that most of VENUS is not to be changed (design goal 1).
3. VENUS should be able to use any QM code in addition to NWChem. This goal requires that any calls from VENUS are not direct calls to NWChem, but to a VENUS interface. This VENUS interface can then be generalized for any QM code or for no QM code. In the latter case, stub routines can be generated to give reasonable results when there is a lack of QM code available.

Keeping these goals in mind, the general model used within VENUS is to initiate the parallel environment using the GA functionality within NWChem, determine which process has the 0 or master identity, have most of the VENUS code run in a sequential manner on process 0, broadcast information to the other processes, and use the other processes only when a QM computation is required. The following describes, in detail, the execution of the VENUS/NWChem coupled software which is shown schematically in Fig. 2.

VENUS is the code that propagates the direct dynamics trajectories and, therefore, is the program that drives the overall execution of VENUS/NWChem. VENUS remains a sequential code which only runs on the master node (design goal 1). As the first executable line in Venus, the new *VENUS_parallel_init* subroutine is called to set up the parallel environment using the *net_init* routine of NWChem that returns the identifier of the process (*myid*) and the total number of processes (*nprocs*). (For a parallel computing architecture it is possible, but not usually desirable, to run more than one process on a given core). Note that if VENUS is not running with NWChem linked, a stub routine is linked in that returns *myid* = 0 and *nprocs* = 1 (design goal 3). (It is important to note that this call is not a direct NWChem call and, if desired, in future work the software interface developed here may be used to link VENUS with other electronic structure codes). Once *myid* and *nprocs* have been initialized, VENUS proceeds to initializing arrays and parameters. All processes are involved in the initialization so that all of the processes are set up correctly and, again, so that as little interference as possible with VENUS is achieved (design goal 1).

However, only the master process reads the VENUS input file since parallel reading of one file is time consuming and can, in principle, cause errors. This is accomplished by a simple if-then

construct that checks if *myid* = 0 around the reading of the input file. All of the input is read at the beginning of the computation, which has the effect of minimizing the changes in VENUS. The VENUS input file describes whether a computation will require a QM code. If so, the variable *NMO* will be set to greater than 0 and the file will also contain basic information for the QM run (e.g. theory type and basis set) that needs to be passed along to the QM code. At the end of the if-then, the VENUS routine *broadcast_data* is called by all processes to broadcast *NMO* from the master process to the rest of the processes. This broadcast also acts as a synchronization point where none of the processes will continue until all of the processes have executed the broadcast. When NWChem is being used, *broadcast_data* will call the NWChem interface function *net_broadcast* to broadcast the information. Once all processes know that VENUS will require a QM computation, all of the processes except the master process call *VENUS_parallel_loop*. This function is an infinite loop with a call to *VENUS_parallel_idop* inside of it. *VENUS_parallel_idop* is the routine where the processes wait in a broadcast for the master to tell them what type of calculation to perform. Note that by placing the other processes in this infinite loop, they do not enter the rest of the VENUS code execution directly, minimizing any other changes in VENUS (design goal 1).

In the meantime, the master process continues with the setup of the computation for VENUS. Towards the end of that setup, the master will have enough information to initialize the QM code. At this point, the master will call *VENUS_parallel_idop* and “catch up” to the other processes. Just before entering this routine, the master will set an operation identification (*idop*) that identifies what type of operation all of the processes are required to perform. The current values for *idop* are given in Table 1 which also includes the mapping of the *idop* value to a computational operation, and the VENUS and NWChem calls that executed with each of the operations.

For this first call, *idop* is set equal to 1 to tell everyone to initialize the QM code. Once *idop* has been broadcast to all of the processes in *VENUS_parallel_idop*, *VENUS_QM_init* (the generalized interface to VENUS for the initialization of QM software) is called which will in turn call the NWChem *initialize_qm* routine. Once all processes have completed this initialization, the master goes back into the VENUS code execution, while others wait in the broadcast in *VENUS_parallel_idop*. This execution and wait mode of operation is continued for all processes except the master process until the QM finalization is called.

The master process will make decisions such as whether a Hessian will be calculated to obtain initial conditions for the computation and what geometry will be used next for an energy and gradient computation. The trajectory is calculated (i.e. coordinates and momenta/velocities updated) by numerically integrating, within VENUS, either Hamilton’s or Newton’s classical equations of motion. The numerical integration is initiated with the potential energy and gradient (and also the Hessian for some integration algorithms) for the initial set of coordinates. This information is used to update the coordinates and momenta/velocities to obtain their new (current) values. The code then enters the *VENUS_parallel_idop* with *idop* = 3 to get the energy and gradient (and the code possibly enters the *VENUS_parallel_idop* with *idop* = 6 to get the Hessian if needed) for this current set of coordinates using the routines laid out in Table 1. This calculation takes most of the computation time. After each integration step VENUS tests for conclusion of the trajectory, which is defined by propagation of the trajectory for a certain number of integration steps or the geometry of the reactive system has attained a certain criterion (e.g. a specific atom–atom separation for product formation). Once the trajectory is complete, the master will send *idop* with a value of 5 and all of the processes will call *VENUS_parallel_final* which in turn will call NWChem’s *finalize_qm* and *net_final* to clean up NWChem’s envi-

Master Node	Children Nodes				
0	1	2	...	(nnodes-1)	Myid
<i>VENUS_parallel_init</i>	<i>VENUS_parallel_init</i>	<i>VENUS_parallel_init</i>	Interface
	Initialize Arrays				
Read Input File					VENUS
<i>broadcast_data</i>	<i>broadcast_data</i>	<i>broadcast_data</i>	Interface
	Synchronization				
More Setup in VENUS					VENUS
<i>VENUS_parallel_idop (idop=1)</i>	<i>VENUS_parallel_idop</i>	<i>VENUS_parallel_idop</i>	Interface
	Initialize QM code				NWChem
Direct Dynamics, Updating Geometry					VENUS
<i>VENUS_parallel_idop (idop=3)</i>	<i>VENUS_parallel_idop</i>	<i>VENUS_parallel_idop</i>	Interface
	Calculate Energy, Gradient and Hessian if Necessary				NWChem
Finalization detection					VENUS
Repeat Last 4 Steps if Needed					VENUS
<i>VENUS_parallel_idop (idop=5)</i>	<i>VENUS_parallel_idop</i>	<i>VENUS_parallel_idop</i>	Interface
	Finalize QM Code				NWChem
Clean up and Stop					VENUS

Fig. 2. Schematic representation of the execution of VENUS/NWChem. The light blue background represents processes that are running and the light gray background represents processes that are waiting. The subroutines are in italic (black for those running on the master node and red for those running on the children nodes). Functions of the interface are in bold (when all the nodes run together) or normal (when only the master node runs).

Table 1

Mapping of idop to a computational operation and the VENUS and NWChem routines.

idop	Computation	VENUS routine	NWChem routine
1	QM initialization	<i>VENUS_QM_init</i>	<i>initialize_qm</i>
2	Gradient	<i>VENUS_QM_gradient</i>	<i>evaluateGradient</i>
3	Energy and gradient	<i>VENUS_QM_gradient_energy</i>	<i>evaluateObjectiveAndGradient</i>
4	Energy	<i>VENUS_QM_energy</i>	<i>evaluateObjective</i>
5	QM finalization	<i>VENUS_parallel_final</i>	<i>finalize_qm</i> and <i>net_final</i>
6	Hessian	<i>VENUS_QM_hessian</i>	<i>evaluateHessian</i>

ronment. The final results of the trajectory, such as the reaction pathway, the product energy partitioning, velocity scattering angle, number of TS recrossings, etc. are determined by VENUS. After these analyses all processes can then stop execution and the job is complete, and all allocated memory can be released. The above description of the VENUS/NWChem interface is shown schematically in Fig. 2.

5. Example usage

The VENUS/NWChem software package has been used to study bimolecular and unimolecular reactions, post-transition state dynamics, and development of algorithms to enhance direct dynamics simulations. The specific reactions and dynamics which have been investigated are: post-transition state dynamics for propene ozonolysis, and intramolecular and unimolecular dynamics of molozonide [37]; development of Hessian-based predictor-corrector integration algorithms for direct dynamics simulations [12]; bimolecular and post-transition state dynamics of the $F^- + CH_3OOH$ reaction [38]; bimolecular dynamics of the $Cl^- + CH_3I \rightarrow ClCH_3 + I^-$ reaction [39]; unimolecular dynamics of the twist-boat intermediate in cyclohexane isomerization [40]; reaction dynamics of $NH_4^+ + CH_4$ collisions [41]; development of Hessian updating algorithms [13]; bimolecular dynamics of the $F^- + CH_3I \rightarrow FCH_3 + I^-$ reaction [42,43]; post-transition state dynamics of carbocation rearrangements on a bifurcating potential energy surface [44,45]; thermal unimolecular decomposition of the epoxy resin constituent $CH_3-NH-CH=CH-CH_3$ [46,47]; $C_2H_5 \rightarrow H + C_2H_4$ unimolecular dynamics [48]; dioxetane unimolecular dissociation, and post-transition state dynamics follow-

ing $^3O_2 + C_2H_4 \rightarrow \cdot O-O-CH_2-CH_2\cdot$ association and an electronic non-adiabatic transition [49]; bimolecular dynamics of the $OH^- + CH_3I$ and $OH^-(H_2O) + CH_3I$ reactions [50]; and calculation of vibrational spectra by the semiclassical-IVR algorithm [51,52].

The parallelism with VENUS/NWChem arises from the parallel computing already implemented in NWChem and the trivial simultaneous launching of multiple direct dynamics trajectories. VENUS is currently not a parallel code, but some initial steps have been taken to parallelize the MM algorithms in VENUS [53]. Since the direct dynamics trajectories may have considerably different atomistic motions, including mechanisms, and thus require different times to complete, it is better use of compute time to launch the trajectories independently and not as one job. Quantitative descriptions of the parallel computing in NWChem have been given previously in Ref. [23].

6. Summary

This paper describes the tight coupling of VENUS and NWChem for the VENUS/NWChem software package, which enables QM, QM+MM, and QM/MM direct dynamics simulations [10]. With the algorithms in VENUS for selecting initial conditions for classical chemical dynamics simulations, VENUS/NWChem may be used to study the dynamics of unimolecular [54] and bimolecular reactions [55], gas-surface collisions [2], and post-transition state dynamics [56]. Work is in progress to enhance the VENUS/NWChem software package so that it may be used to study liquid phase chemical reaction dynamics [57]. VENUS/NWChem enables a wide variety of direct chemical dynamics simulations to: interpret experimental results and understand the atomic-level

dynamics of chemical reactions; illustrate the ability of classical simulations to correctly interpret and predict chemical dynamics, when quantum effects are expected to be unimportant; obtain the correct classical dynamics predicted by an electronic structure theory; acquire a deeper understanding of when statistical theories are valid for predicting the mechanisms and rates of chemical reactions; and discover new reaction pathways and chemical dynamics. A substantial number of simulations are expected to be completed with the VENUS/NWChem software package.

Acknowledgments

The development and applications of the VENUS/NWChem software package by the Hase Research Group have been supported by grants from the Air Force Office of Scientific Research, the Office of Naval Research, and the National Science Foundation. Support from the Robert A. Welch Foundation, from Grant No. D-0005, is also important. This material is also based upon work supported by the National Science Foundation under Grant No. OISE-0730114 for the Partnerships in International Research and Education (PIRE). This work was done in part using EMSL, a national scientific user facility sponsored by the Department of Energy's Office of Biological and Environmental Research and located at Pacific Northwest National Laboratory, operated for the U.S. Department of Energy by Battelle under contract DE-AC05-76RL01830.

The infrastructure of the High Performance Computing Center (HPCC) at Texas Tech University, under the direction of Philip W. Smith, is critical for this research.

References

- [1] W.L. Hase, D.M. Ludlow, R.J. Wolf, T. Schlick, *J. Phys. Chem.* **85** (1981) 958.
- [2] D.G. Schultz, S.B. Weinhaus, L. Hanley, P. de Sainte Claire, W.L. Hase, *J. Chem. Phys.* **106** (1997) 10337.
- [3] W.L. Hase, R.J. Wolf, C.S. Sloane, *J. Chem. Phys.* **71** (1979) 2911; *J. Chem. Phys.* **76** (1982) 2771 (erratum).
- [4] D.L. Bunker, W.L. Hase, *J. Chem. Phys.* **59** (1973) 4621; *J. Chem. Phys.* **69** (1978) 4711 (erratum).
- [5] D.-H. Lu, W.L. Hase, R.J. Wolf, *J. Chem. Phys.* **85** (1986) 4422.
- [6] P. de Sainte Claire, K.C. Hass, W.F. Schneider, W.L. Hase, *J. Chem. Phys.* **106** (1997) 7331.
- [7] K. Bolton, W.L. Hase, C. Doubleday Jr., *J. Phys. Chem. B* **103** (1999) 3691.
- [8] D.L. Bunker, *Methods Comput. Phys.* **10** (1971) 287.
- [9] G.H. Peshlherbe, H. Wang, W.L. Hase, *Adv. Chem. Phys.* **105** (1999) 171.
- [10] L. Sun, W.L. Hase, *Rev. Comput. Chem.* **19** (2003) 79.
- [11] J.M. Millam, V. Bakken, W. Chen, W.L. Hase, H.B. Schlegel, *J. Chem. Phys.* **111** (1999) 3800.
- [12] U. Lourderaj, K. Song, T.L. Windus, Y. Zhuang, W.L. Hase, *J. Chem. Phys.* **126** (2007) 044105.
- [13] H. Wu, M. Rahman, J. Wang, U. Lourderaj, W.L. Hase, Y. Zhuang, *J. Chem. Phys.* **133** (2010) 074101.
- [14] U. Burkert, N.L. Allinger, *Molecular Mechanics*, in: ACS Monograph, vol. 177, American Chemical Society, Washington, DC, 1982.
- [15] W.L. Hase, G. Mrowka, R.J. Brudzynski, C.S. Sloane, *J. Chem. Phys.* **69** (1978) 3548; *J. Chem. Phys.* **72** (1980) 6321 (erratum).
- [16] S.R. Vande Linde, W.L. Hase, *J. Phys. Chem.* **94** (1990) 2778.
- [17] I.S.Y. Wang, M. Karplus, *J. Amer. Chem. Soc.* **95** (1973) 8160.
- [18] C. Leforestier, *J. Chem. Phys.* **68** (1978) 4406.
- [19] T.N. Truong, D. Lu, G.C. Lynch, Y.P. Liu, V.S. Melissas, J.J.P. Stewart, R. Steckler, B.C. Garrett, A.D. Isaacson, A. Gonzalez-Lafont, N.R. Sachchida, G.C. Hancock, T. Joseph, D.G. Truhlar, *Comput. Phys. Comm.* **75** (1) (1993) 143.
- [20] J. Zheng, S. Zhang, J.C. Corchado, Y.-Y. Chuang, E.L. Coitino, B.A. Ellingson, D.G. Truhlar, GAUSSRATE-Version 2009-a, University of Minnesota, Minneapolis, 2010.
- [21] J. Zheng, M.A. Iron, B.A. Ellingson, J.C. Corchado, Y.-Y. Chuang, D.G. Truhlar, NWCHEMATE-Version 2007, University of Minnesota, Minneapolis, 2007.
- [22] J. Zheng, D.G. Truhlar, JAGUARATE-Version 2007, University of Minnesota, Minneapolis, 2007.
- [23] M. Valiev, E.J. Bylaska, D. Wang, K. Kowalski, N. Govind, T.P. Straatsma, J. Nieplocha, E. Aprà, T.L. Windus, W.A. de Jong, *Comput. Phys. Comm.* **181** (2010) 1477.
- [24] M.S. Gordon, M.W. Schmidt, in: C.E. Dykstra, G. Frenking, K.S. Kim, G.E. Scuseria (Eds.), *Advances in Electronic Structure Theory: GAMESS A Decade Later*, in: *Theory Appl Comput Chem: First Forty Years*, Elsevier, Amsterdam, 2005, p. 1167.
- [25] H.-J. Werner, P.J. Knowles, G. Knizia, F.R. Manby, M. Schütz, P. Celani, T. Korona, R. Lindh, A. Mitrushenkov, G. Rauhut, K.R. Shamasundar, T.B. Adler, R.D. Amos, A. Bernhardsson, A. Berning, D.L. Cooper, M.J.O. Deegan, A.J. Dobson, F. Eckert, E. Goll, C. Hampel, A. Hesselmann, G. Hetzer, T. Hrenar, G. Jansen, C. Köppl, Y. Liu, A.W. Lloyd, R.A. Mata, A.J. May, S.J. McNicholas, W. Meyer, M.E. Mura, A. Nicklass, D.P. O'Neill, P. Palmieri, P. Pflüger, R. Pitzer, M. Reiher, T. Shiozaki, H. Stoll, A.J. Stone, R. Tarroni, T. Thorsteinsson, M. Wang, A. Wolf, MOLPRO, version 2010.1. A package of ab initio programs, see <http://www.molpro.net>.
- [26] W.L. Hase, R.J. Duchovic, X. Hu, A. Komornicki, K.F. Lim, D. Lu, G.H. Peshlherbe, K.N. Swamy, S.R. Vande Linde, A. Varandas, J. Wang, R.J. Wolf, *Bull. Quant. Chem. Program Exchange (QCPE)* **16** (1996) 671.
- [27] X. Hu, W.L. Hase, T.J. Pirraglia, *J. Comput. Chem.* **12** (1991) 1014.
- [28] D.L. Bunker, *Methods Comput. Phys.* **10** (1971) 287.
- [29] T. Schlick, *Molecular Modeling and Simulation*, Springer, NY, 2000.
- [30] J. Nieplocha, B. Palmer, V. Tipparaju, M. Krishnan, H. Trease, E. Apra, *Int. J. High Perf. Comput. Appl.* **20** (2006) 203.
- [31] J. Nieplocha, V. Tipparaju, M. Krishnan, D. Panda, *Int. J. High Perf. Comput. Appl.* **20** (2006) 233.
- [32] Y. Alexeev, B.A. Allan, R.C. Armstrong, D.E. Bernholdt, T.L. Dahlgren, D. Gannon, C.L. Janssen, J.P. Kenny, M. Krishnan, J.A. Kohl, G. Kumfert, L. Curfman McInnes, J. Nieplocha, S.G. Parker, C. Rasmussen, T.L. Windus, *J. Phys. Conf. Ser.* **16** (2005) 536.
- [33] C.L. Janssen, I.B. Nielsen, M.L. Leininger, E.F. Valeev, J.P. Kenny, E.T. Seidl, *The Massively Parallel Quantum Chemistry Program (MPQC), Version III*, Sandia National Laboratories, Livermore, CA, USA, 2008.
- [34] F. Peng, M.S. Wu, M. Sosonkina, T.L. Windus, J. Bentz, M.S. Gordon, J. Kenny, C.L. Janssen, *Proc. of HPC-GECO/CompFrame 2007* (2007) 101.
- [35] J.P. Kenny, S.J. Benson, Y. Alexeev, J. Sarich, C.L. Janssen, L. Curfman McInnes, M. Krishnan, J. Nieplocha, E. Jurrus, C. Fahlstrom, T.L. Windus, *J. Comput. Chem.* **25** (2004) 1717.
- [36] T.P. Gulabani, M. Sosonkina, M.S. Gordon, C.L. Janssen, J.P. Kenny, H. Netzloff, T.L. Windus, *Proceedings of the 2009 Spring Simulation Multiconference* (San Diego, California, March 22–27, 2009). Society for Computer Simulation International, San Diego, CA, pp. 1–6.
- [37] G. Vayner, S.V. Addepalli, K. Song, W.L. Hase, *J. Chem. Phys.* **125** (2006) 014317.
- [38] J.G. López, G. Vayner, U. Lourderaj, S.V. Addepalli, S. Kato, W.A. de Jong, T.L. Windus, W.L. Hase, *J. Amer. Chem. Soc.* **129** (2007) 9976.
- [39] J. Mikosch, S. Trippel, C. Eichhorn, R. Otto, U. Lourderaj, J.X. Zhang, W.L. Hase, M. Weidemüller, R. Wester, *Science* **319** (2008) 183.
- [40] K. Kakhiani, U. Lourderaj, W. Hu, D.M. Birney, W.L. Hase, *J. Phys. Chem. A* **113** (2009) 4570.
- [41] G.L. Barnes, W.L. Hase, *J. Phys. Chem. A* **113** (2009) 7543.
- [42] J. Zhang, J. Mikosch, S. Trippel, R. Otto, M. Weidemüller, R. Wester, W.L. Hase, *J. Phys. Chem. Lett.* **1** (2010) 2747.
- [43] J. Mikosch, J. Zhang, S. Trippel, R. Otto, R. Sun, W.A. de Jong, M. Weidemüller, W.L. Hase, R. Wester, *J. Amer. Chem. Soc.* **135** (2013) 4250.
- [44] M.R. Siebert, J. Zhang, S.V. Addepalli, D.J. Tantillo, W.L. Hase, *J. Amer. Chem. Soc.* **133** (2011) 8335.
- [45] M.R. Siebert, P. Manikandan, R. Sun, D.J. Tantillo, W.L. Hase, *J. Chem. Theory Comput.* **8** (2012) 1212.
- [46] L. Yang, R. Sun, W.L. Hase, *J. Chem. Theory Comput.* **7** (2011) 3478.
- [47] L. Yang, R. Sun, W.L. Hase, *Comput. Theoret. Chem.* **990** (2012) 62.
- [48] P. Manikandan, W.L. Hase, *J. Chem. Phys.* **136** (2012) 184110.
- [49] R. Sun, K. Park, W.A. de Jong, H. Lischka, T.L. Windus, W.L. Hase, *J. Chem. Phys.* **137** (2012) 044305.
- [50] R. Otto, J. Xie, J. Brox, S. Trippel, M. Stei, T. Best, M.R. Siebert, W.L. Hase, R. Wester, *Faraday Discuss.* **157** (2012) 41.
- [51] Y. Zhuang, M.R. Siebert, W.L. Hase, K.G. Kay, M. Ceotto, *J. Chem. Theory Comput.* **9** (2013) 54.
- [52] M. Ceotto, Y. Zhuang, W.L. Hase, *J. Chem. Phys.* **138** (2013) 054116.
- [53] R. Rajagopalan, *Parallelizing the Calculation of a Classical Trajectory*, Master Thesis, Texas Tech University, 2007.
- [54] U. Lourderaj, W.L. Hase, *J. Phys. Chem. A* **113** (2009) 2236.
- [55] P. Manikandan, J. Zhang, W.L. Hase, *J. Phys. Chem. A* **116** (2012) 3061.
- [56] U. Lourderaj, K. Park, W.L. Hase, *Int. Rev. Phys. Chem.* **27** (2008) 361.
- [57] S. Kohale, W.L. Hase, in preparation.