The Journal of Chemical Physics SOFTWARE pubs.aip.org/aip/jcp

An efficient and universal parallel algorithm for high-dimensional quantum dynamics in poly-atomic reactions

Cite as: J. Chem. Phys. 160, 202502 (2024); doi: 10.1063/5.0209245
Submitted: 19 March 2024 • Accepted: 6 May 2024 •
Published Online: 24 May 2024

Yong Zhou,[1,2,a)] Yunpeng Lu,[3] Zhaojun Zhang,[2,a)] and Dong H. Zhang[2,4]

AFFILIATIONS

[1] Anhui Province Key Laboratory for Control and Applications of Optoelectronic Information Materials, Department of Physics, Anhui Normal University, Wuhu 241000, People's Republic of China
[2] State Key Laboratory of Molecular Reaction Dynamics and Center for Theoretical and Computational Chemistry, Dalian Institute of Chemical Physics, Chinese Academy of Sciences, Dalian 116023, People's Republic of China
[3] School of Chemistry, Chemical Engineering and Biotechnology, Nanyang Technological University, Singapore 637371, Singapore
[4] Hefei National Laboratory, Hefei 230088, People's Republic of China

[a)] **Authors to whom correspondence should be addressed:** yong.zhou@mail.ahnu.edu.cn and zhangzhj@dicp.ac.cn

ABSTRACT

This study presents a parallel algorithm for high-dimensional quantum dynamics simulations in poly atomic reactions, integrating distributed- and shared-memory models. The distributions of the wave function and potential energy matrix across message passing interface processes are based on bundled radial and angular dimensions, with implementations featuring either two- or one-sided communication schemes. Using realistic parameters for the H + NH_3 reaction, performance assessment reveals linear scalability, exceeding 90% efficiency with up to 600 processors. In addition, owing to the universal and concise structure, the algorithm demonstrates remarkable extensibility to diverse reaction systems, as demonstrated by successes with six-atom and four-atom reactions. This work establishes a robust foundation for high-dimensional dynamics studies, showcasing the algorithm's efficiency, scalability, and adaptability. The algorithm's potential as a valuable tool for unraveling quantum dynamics complexities is underscored, paving the way for future advancements in the field.

I. INTRODUCTION

A bimolecular reaction transpires due to the collision between two reactant molecules, generating novel product molecules.[1] Understanding the behaviors at the quantum level is fundamental to unraveling the mysteries of chemical reactions. Quantum dynamics simulations, a powerful subset of computational chemical physics, provide a significant means to explore and comprehend the intricate processes that govern molecular interactions.[2]

Tremendous progress has been witnessed in quantum dynamics calculations on chemical reactions. The first full-dimensional quantum studies of the simplest H + H_2 reaction[3,4] were achieved in the 1970s, with time-independent (TID) quantum dynamics methods.[5,6] Later on, quantum dynamics were extended to larger-sized reactions comprising more atoms, where the extension of TID methods became quite tricky. Accurate initial-state-selected (ISS) time-dependent wave packet (TDWP) calculations of the H_2 + OH reaction were pioneered by Zhang and Zhang[7] and Neuhauser[8] in 1994 and of the H + H_2O reaction by Zhang and Light[9] in 1996. Since then, reactions with five or more atoms have attracted increasing interest in the field.[10,11]

An accurate quantum simulation for such poly-atomic reactions often suffers from the overwhelming amount of computational resources required. As a result, various approximate reduced-dimensional models were usually adopted to balance the simulation accuracy and computation workload. For the typical five-atom reaction of H + NH_3, a four-dimensional semi-rigid vibrating rotor target (SVRT) model[12] and a seven-dimensional dynamics model[13] were proposed. For the prototype six-atom reaction of H + CH_4, simplified models with up to ten dimensions were developed.[14–17]

Developments in full-dimensional calculations for poly-atomic reactions have also been seen in the past two decades. Yang studied the five-atom H + NH_3 reaction[18] using the TDWP method mentioned above. Alternatively, a new transition-state wave packet (TSWP) method for the reactive scattering was proposed[19] and applied to atom–diatom reactions[20] and some reactions beyond three atoms.[21,22] Notably, within the less "brute-force"[23] multiconfiguration time-dependent Hartree (MCTDH)[24,25] framework, the group of Manthe and co-workers has played a pioneering role in simulating reaction dynamics of systems with six atoms.[26–32] However, the grids and basis used in these studies were suspected to be less than required by a convergency of simulation results, especially for high-energy regions,[33] leading naturally to more or less accuracy losses. This dilemma is primarily due to the unsatisfactory efficiency of relevant computer programs instead of the availability and accessibility of computational resources. Thus, further efforts are still desirable in developing high-performance software for high-dimensional quantum dynamics simulations to deeply tap into the potential of modern computers.

The computational challenge of large-scale quantum calculations, requiring a tremendous amount of computer resources, such as CPU time and physical memory, is very suitable for parallel computing, particularly in the case of the time-dependent wave packet (TDWP) propagations consisting of an enormous amount of inherent parallelism.[2] Goldfield and Gray proposed distributing all Ω quantum numbers (projection of total angular momentum J on the body-fixed axis) over different computer processors.[34] Cvitas and Althorpe replaced the angular kinetic energy part of the split-operator propagator with the Crank–Nicholson approximation, making a more straightforward parallel Coriolis-coupled quantum dynamics.[35] Eggert *et al.* implemented a parallel pseudo-spectral calculation of molecular energy levels by splitting the evaluation of the potential term in Hamiltonian over up to 64 computer processors.[36] Goldfield presented a parallel method for acting the Hamiltonian for four-atom quantum dynamics calculations, where the data were distributed to processors according to one radial coordinate.[37] Similarly, Meijer reported another parallel strategy with the wave function distributed according to two degrees of freedom.[38] Borowski and co-workers introduced a parallel implementation for quantum dynamics of molecule-surface reaction with problem-adapted data decomposition in all dimensions.[39] Medvedev, Goldfield, and Gray[40] presented accelerating the four-atom quantum mechanics by a hybrid OpenMP/MPI approach, in which the wave function was distributed over a two-dimensional (2D) radial domain formed with the selected degrees of freedom, realizing good scalability for up to 512 processors. Note that parallelization of the MCTDH method was also investigated previously for shared-memory[41] and distributed-memory[42] hardware, which is out of the scope of our current work.

Despite a wealth of successful investigations in parallelizing quantum dynamics codes for reactions consisting of up to four atoms, to the best of the authors' knowledge, a report of parallel algorithms, scalable to full-dimensional quantum TDWP simulation for reactions with more atoms, is still seldom in the literature. This paper presents a novel approach to parallel quantum time-dependent wave packet calculation, exploiting the inherent parallelism common among reactions with different numbers of atoms. Our approach scales very well up to hundreds of processors, which showed its power in the full-dimensional quantum studies of the H + NH_3 (basis size: about two billion)[43] and H + CH_4 (basis size: roughly one hundred billion)[33] reactions, in addition to the H + D_2O reaction.[44] Due to the algorithm's universal idea and concise structure, the effort needed to extend it to other reaction systems is trivial.

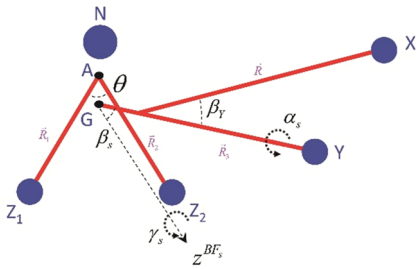

FIG. 1. Jacobi and Radau vectors for the X + YNZ_2 system.

This article is organized as follows: Section II outlines the time-dependent wave packet method with the five-atom X + YNZ_2 reaction as an example. Section III describes the parallel implementation of the corresponding time-dependent wave packet program. Section IV presents performance tests for realistic simulations. A summary concludes the article.

II. TDWP METHOD FOR THE X + YNZ_2 REACTION

The five-atom reaction system X + $YNZ_{1,2}$ is described by a combination of Jacobi and Radau vectors,[43,45,46] as shown in Fig. 1. The Radau vectors, $\vec{R}_i$ (i = 1, 2), are the vectors $\overrightarrow{AZ_i}$, with A being the "canonical" point of $NZ_{1,2}$. Thus, the internal vibration of $NZ_{1,2}$ is described by three spherical coordinates: R_2, R_1, and θ. The Jacobi vector $\vec{R}_3$ is from the center mass of $NZ_{1,2}$ to atom Y, and $\vec{R}$ is from the center mass to atom X. The body-fixed frame of $YNZ_{1,2}$ ($BF_{YNZ1,2}$) is oriented by Euler angles with respect to the global body-fixed frame (BF) of the whole system (X + $YNZ_{1,2}$), β_Y and α_s. The orientation of $NZ_{1,2}$'s body-fixed frame (BF_s) is parametrized by Euler angles β_s and γ_s with respect to the $BF_{YNZ1,2}$ frame.

As a result, the Hamiltonian of the reactant X + $YNZ_{1,2}$ can be written as ($\hbar$ = 1)

$$\begin{aligned}\hat{H} = &-\frac{1}{2\mu_R}\frac{\partial^2}{\partial R^2} + \frac{(\vec{J}^{\dagger}_{tot} - \vec{L}^{\dagger}_{YNZ_{1,2}})\cdot(\vec{J}_{tot} - \vec{L}_{YNZ_{1,2}})}{2\mu_R R^2} \\ &- \frac{1}{2\mu_{R_3}}\frac{\partial^2}{\partial {R_3}^2} + \frac{(\vec{L}^{\dagger}_{YNZ_{1,2}} - \vec{L}^{\dagger}_{NZ_{1,2}})\cdot(\vec{L}_{YNZ_{1,2}} - \vec{L}_{NZ_{1,2}})}{2\mu_{R_3} R_3^2} \\ &+ \hat{T}_{NZ_{1,2}} + V(R, \beta_Y, \alpha_s, R_3, \beta_s, \gamma_s, R_1, R_2, \theta), \end{aligned} \quad (1)$$

where μ_R and μ_{R_3} are the reduced masses of X-$YNZ_{1,2}$ and Y-$NZ_{1,2}$, respectively. $\vec{J}_{tot}$ is the total angular momentum operator of the system. $\vec{L}_{YNZ_{1,2}}$ and $\vec{L}_{NZ_{1,2}}$ are the rotational angular momentum operator of the $YNZ_{1,2}$ and $NZ_{1,2}$, respectively. V denotes the potential energy surface.

The kinetic operator $\hat{T}_{NZ_{1,2}}$ include three parts, $\hat{T}_{NZ_{1,2}} = \hat{T}_{vib} + \hat{T}_{rot} + \hat{T}_{cor}$, with $\hat{T}_{vib}$ being the vibrational part of $NZ_{1,2}$, $\hat{T}_{rot}$ being

the rotational part of $NZ_{1,2}$, and $\hat{T}_{corr}$ being the coupling between the rotation and vibration in $NZ_{1,2}$. The $\hat{T}_{vib}$ operator is defined by

$$2\hat{T}_{vib} = -\sum_{i=1,2}\frac{1}{\mu_i}\frac{\partial^2}{\partial R_i^2} - \left(\frac{1}{\mu_1 R_1^2} + \frac{1}{\mu_2 R_2^2}\right)\frac{\partial}{\partial u}v^2\frac{\partial}{\partial u}, \quad (2)$$

where $\mu_1 = m_{Z_1}$, $\mu_2 = m_{Z_2}$, $u = \cos\theta$, and $v = \sin\theta$. Explicit forms for the other operators, $\hat{T}_{rot}$ and $\hat{T}_{corr}$, can be found in the literature.[43]

During the wave packet propagation, the nine-dimensional wave function is expanded as

$$\Psi_{vL}^{J_{tot}MK}(\vec{R},\vec{R}_3,\vec{R}_1,\vec{R}_2,t) = \sum_{\substack{R,R_3,R_1,R_2,\\ \theta,\beta_s,\gamma_s,L,k,K}} F_{R,R_1,R_2,R_3,\theta,\beta_s,\gamma_s,L,k}^{J_{tot}MK} \times u_n^{v_{R_3}}(R)\phi_{v_{R_3}}(R_3)\phi_{R_1}(R_1)\phi_{R_2}(R_2) \times \phi_\theta(\theta)\phi_{\beta_s}(\beta_s)\phi_{\gamma_s}(\gamma_s)Y_{Lk}^{J_{tot}MK}(\vec{R},\vec{R}_3), \quad (3)$$

where n is the translational basis label and (vL) is the initial rovibrational state of $YNZ_{1,2}$. M and K are the projection of the total angular momentum J_{tot} on the z axis of space-fixed and body-fixed frames, respectively. In addition, k is the projection of the rotational angular momentum L on the z axis of the $YNZ_{1,2}$ frame.

In order to solve the Schrödinger equation, the potential-optimized discrete variable representation (PODVR) is used for six degrees of freedom $(R_1, R_2, R_3, \theta, \beta_s, \gamma_s)$, leading to the corresponding basis sets, i.e., $\phi_{R_1}(R_1)$, $\phi_{R_2}(R_2)$, $\phi_{R_3}(R_3)$, $\phi_\theta(\theta)$, $\phi_{\beta_s}(\beta_s)$, and $\phi_{\gamma_s}(\gamma_s)$. Here, $u_n^{v_{R_3}}(R)$ represents the basis set functions for the reactive coordinate, R, which are defined separately for grid ranges of the asymptotic and interaction region, respectively.

The rotational basis functions for the X + $YNZ_{1,2}$ system in the BF frame are written as

$$Y_{Lk}^{J_{tot}MK}(\vec{R},\vec{R}_3) = \overline{D}_{MK}^{J_{tot}}(\vec{R})\overline{D}_{Kk}^{L}(\vec{R}_3), \quad (4)$$

where $\overline{D}_{MK}^{J}(\vec{R})$ and $\overline{D}_{Kk}^{L}(\vec{R}_3)$ are the Wigner rotation matrices, depending on the corresponding Euler angles $(\alpha,\beta,\gamma,\beta_Y,\alpha_s)$ between the coordinate frames.

For the specific (J_{tot}, M), a direct product of a localized translational wave packet $G(R)$ and a specific rovibrational eigenstate $(v_0L_0K_0)$ for $YNZ_{1,2}$ is chosen to construct the initial wave function,

$$\Psi_{v_0L_0}^{JMK_0}(\vec{R},\vec{R}_3,\vec{R}_1,\vec{R}_2,\ t=0) = G(R)\psi_{v_0L_0}^{JMK_0}(\vec{R},\vec{R}_3,\vec{R}_1,\vec{R}_2), \quad (5)$$

where $\psi_{v_0L_0}^{JMK_0}(\vec{R},\vec{R}_3,\vec{R}_1,\vec{R}_2)$ is the eigenfunction of $YNZ_{1,2}$, calculated when the molecular $YNZ_{1,2}$ is far from the incoming atom X.

The initial wave packet is propagated using the second-order split operator method,[47]

$$\Psi(t+\Delta) = e^{-i\hat{T}_{rad}\Delta/2}e^{-i\hat{T}_{ang}\Delta/2}e^{-i\hat{U}\Delta}e^{-i\hat{T}_{ang}\Delta/2}e^{-i\hat{T}_{rad}\Delta/2}\Psi(t), \quad (6)$$

where the radial kinetic energy operator $\hat{T}_{rad} = -\frac{1}{2\mu_R}\frac{\partial^2}{\partial R^2} + \sum_{i=1}^{3}\hat{h}_i$ and the angular kinetic energy operator $\hat{T}_{ang} = \hat{H}_0 - \hat{T}_{rad}$, with $\hat{U} = V(\vec{R},\vec{R}_3,\vec{R}_1,\vec{R}_2) - \sum_{i=1}^{3}V_{R_i}^{ref}(R_i)$ and $\hat{H}_0 = \hat{H} - \hat{U}$ as the reference potential and the reference Hamiltonian, respectively.

The time-independent wave function ψ_{iE} is calculated using a Fourier transform of time-dependent of the time-dependent wave function as

$$|\psi_{iE}\rangle = \frac{1}{a_i(E)}\int_0^\infty e^{i(E-H)t/\hbar}|\Psi_i(0)\rangle dt, \quad (7)$$

where $a_i(E) = \langle\phi_{iE}|\Psi_i(0)\rangle$ is the overlap between the initial wave packet $\Psi_i(0)$ and the energy-normalized asymptotic scattering function ϕ_{iE}.

The total reaction probabilities for the specific initial state in the whole range of energies can be obtained by calculating the reactive flux at the dividing surface $R_3 = R_{3s}$ as

$$P_i(E) = \frac{\hbar}{\mu_{R_3}}\mathrm{Im}(\langle\psi_{iE}|\psi'_{iE}\rangle)|_{R_3=R_{3s}}, \quad (8)$$

where ψ'_{iE} the first derivative of the time-independent wave function in R_3.

If anyone is interested in more theoretical details, they can refer to Ref. 43.

III. PARALLEL IMPLEMENTATIONS OF THE TDWP PROGRAM

A. Simulation procedure

The structure of the time-dependent wave packet five-atom nine-dimensional (TDWP-5A9D) program consists of three major stages: initialization, propagation, and finalization. At the initialization stage, the program reads the values of parameters from the input file and then sets the grids for each spatial dimension, matrices for representation transformation, potential energy matrix, and other auxiliary arrays. At the propagation stage, the time-dependent wave packet is evolved using the split-operator method.[47] The integration over time in the Fourier transformation of the time-dependent wave packet defined in Eq. (7) is discretized and performed within each propagation step. At the finalization stage, dynamic information is extracted by analyzing the time-independent wave function. The initialization and finalization stages are done once and for all, which usually occupies only a limited portion of the total calculation time. At the propagation stage, operator multiplications and transformations between different representations are carried out frequently, involving considerable data access over different system dimensions. As a result, the propagation stage is computationally more demanding, especially when one needs to propagate the wave packet for a long time.

B. Wave packet propagation and representation transformations

The wave packet propagation is carried out using the split-operator method in the order shown in Eq. (6). The exponential form of the split-operator method generally requires that the wave function is transformed to a representation in which the matrix for each operator is diagonal before applying the corresponding operator.[36] Specifically, the finite basis representation (FBR) is used for kinetic energy operators, and the discrete variable representation (DVR) is used for the potential energy operator. In the TDWP-5A9D

program, we choose an FBR representation for $\hat{T}_{rad}$, a mixed DVR-FBR representation for $\hat{T}_{ang}$, and a DVR representation for $\hat{U}$, as described below.

The evolution of representations corresponding to each term of the split Hamiltonian is described as follows. At the beginning of each propagation step, the wave packet is in the FBR representation for all the nine dimensions, and the radial kinetic operator $\hat{T}_{rad}$ is applied by half-time step ($\Delta/2$). Then, the four radial dimensions of the wave function are transformed to the corresponding DVR representations (R, R_1, R_2, R_3), and the angular kinetic operator $\hat{T}_{ang}$ is applied. Following that, the angular momentum FBR representation of the wave function is further transformed to the corresponding DVR representations $(\beta_s, \gamma_s, \theta, \beta_Y, \alpha_s)$, and the potential energy operator $\hat{U}$ is applied. Later, the $\hat{T}_{ang}$ and $\hat{T}_{rad}$ operators are applied to the wave function successively in a symmetric manner.

C. Parallel strategy

In the TDWP-5A9D program, we implement a hybrid distributed-memory/shared-memory parallel strategy analogous to that in Ref. 48. The distributed-memory part of the algorithm is based on the message passing interface (MPI),[49] and the share-memory parallel part is based on the OpenMP paradigm.[50] The core idea of the parallel computation underlying the program is "divide and conquer." The most time-consuming wave packet propagation is partitioned and accomplished collaboratively by a batch of computer nodes, each with one MPI process. Most of the calculations are performed locally within each node by exploiting the parallelism based on OpenMP, which is the case when a complete set of data required by a specific operation is entirely stored in the memory of the same node. However, if the data set required is incomplete, i.e., distributed among more than one node, it is necessary to perform inter-process data exchange first, which is achieved through MPI. In the following discussions, the terms "computer node" and "MPI process" are used interchangeably if not specified otherwise.

1. Data distribution

Logically, the entire wave function is thought to consist of two imaginary dimensions, i.e., as a direct product of the "radial" (Rad) and the "angular" (Ang) ones (Rad ⊗ Ang). The length of the "radial" dimension is $nRad = nR^* nR_1{}^* nR_2{}^* nR_3$, with nR and nR_x as the number of basis functions in the FBR representation (or equally grid points in the DVR) for the four actual radial dimensions, respectively. The size of the "angular" dimension in the FBR representation, $nRot$, is the total number of angular momentum basis sets.

Inside the code, the two "imaginary" arrays, which store the full time-dependent wave function (WF) and potential energy (PE), are divided into disjoint parts, with each part assigned to one specific MPI process. The partition scheme of the WF array is shown in Fig. 2(a), where the angular dimension ($nRot$) is divided as evenly as possible, according to the total number of MPI processes $nProc$. Each process maintains an array to store the partial WF, whose dimensions are $nRowp^* nCol$ with $nRowp = nRot/nProc$ and $nCol = nRad$. As a result, the radial kinetic operator $\hat{T}_{rad}$ can be evaluated locally within each MPI process. However, the "angular" dimension is distributed among different MPI processes. In order to locally evaluate the angular kinetic operator $\hat{T}_{ang}$ and the potential energy operator $\hat{U}$, another column-wise partition of the WF and PE arrays is also performed, which can be considered a transpose of that shown in Fig. 2(a). As a result, each process maintains an array to store the partial PE, whose dimensions are $nRow^* nColp$ with $nRow = nAng$ the total number of angular grids and $nColp = nRad/nProc$.

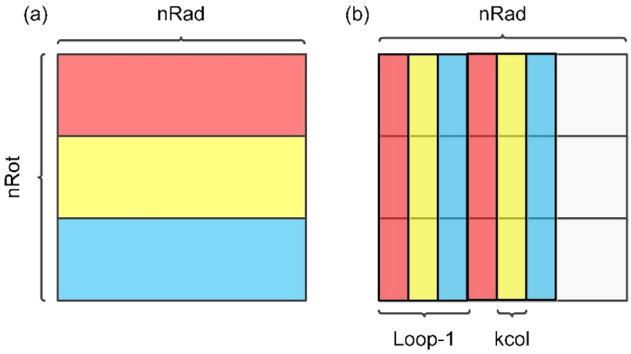

FIG. 2. Partition of the wave function: (a) before and (b) during MPI communication. A total number of three MPI processes are assumed for illustration purposes.

We note that a similar plan, named "Doubly Distribution Parallel ĤΨ" (DDPHP), was proposed previously by Meijer.[38] However, in the DDPHP scheme, the partition of the WF array is performed according to one single dimension or quantum number, which limits the total number of MPI processes available. Alternatively, Medvedev and co-workers proposed partitioning the WF array over a 2D radial domain, e.g., the inter-molecule distance R and the length of breaking bond r_1 in the AB + CD system.[40] Although it may benefit from the less communication between neighboring computer nodes, in practice, an optimal 2D partition of the WF array over MPI processes seems to depend on the exact sizes of the involved dimensions, as shown in the original report, and thus requires careful tunes or expert experiences beforehand, especially for complicated poly atomic reaction systems with many degrees of freedom. By comparison, in our work, the partition of the WF array is generally performed according to the two bundled radial and angular dimensions, which are essentially independent of the reaction system under study, making it easier to implement and apply.

Keep in mind that the column number $nRad$ can be very large sometimes. The huge size of data blocks to be exchanged may lead to inefficient communication, particularly in a busy network environment with limited bandwidth. Accordingly, we introduce a new parameter $kcol$, the maximum column size of a data block allowed to be exchanged each time [Fig. 2(b)]. Consequently, it takes more than one iteration (loop) to evaluate the $\hat{T}_{ang}$ and $\hat{U}$ operations, with the $kcol^* nProc$ columns processed in each loop. Then, the total number of loops over columns is $nLoop = nRad/(kcol^* nProc)$. The introduction of the $kcol$ parameter brings additional flexibility to the simulation, which allows program users to adjust the load of network communications and achieve an optimal overlap between calculation and communication.

2. Inter-process data exchange: Two-sided MPI implement

With the data distribution scheme described above, the full WF array is partitioned row-wise according to the "angular" dimension. Moreover, the full PE array is split column-wise according to the "radial" dimension. The sliced WF and PE arrays are stored in the physical memory of an individual MPI process. Remember that during the wave packet propagation, only the sliced WF array needs to be exchanged among different MPI processes. Here, we would like to name the $\hat{T}_{rad}$ operation evaluated on local MPI processes without data communication as "row operation." In contrast, the evaluations of the $\hat{T}_{ang}$ and $\hat{U}$ operators requiring communication are named "column operations."

An analysis shows that for column operations in one iteration, each MPI process requires three successive steps:

1. Exchanges raw WF data between MPI processes.
2. Does computations, i.e., the actions of $\hat{T}_{ang}$ and $\hat{U}$.
3. Exchanges updated WF data between MPI processes.

In the TDWP-5A9D program, we allocate the following two important arrays:

- **srBuf(2*MAX(nRowp)*$\mathbf{k_{col}}$, 0:nProc-1)**: send/receive buffer storing updated and raw WF data.
- **workTmp(nRot, $\mathbf{k_{col}}$)**: work array holding WF data for $\hat{T}_{ang}$ and $\hat{U}$ evaluations.

In addition, several auxiliary subroutines are also defined,

- **packRawData**: copy local raw data from WF to srBuf;
- **unpackRawData**: arranges raw data in srBuf and stores them in workTmp;
- **packRstData**: copy local updated data from workTmp to srBuf;
- **unpackRstData**: arranges updated data in srBuf and stores them in WF;
- **initExchange**: encapsulates MPI reading/writing utilities and starts to scatter raw/updated data stored in srBuf;
- **finishExchange**: encapsulates the MPI synchronization utility and finalizes data exchanges, guaranteeing that the raw and updated WF data are ready for later use.

The classic non-blocking two-sided MPI utilities (MPI_ISend/MPI_IRecv/MPI_WaitAll) are employed in the first implementation of the parallel algorithm. All these arrays and subroutines defined above are finally assembled into one core subroutine named **ExchangeData**, which is called once for each loop over columns. The corresponding FORTRAN codes are shown in Appendix A 1.

As described in the FORTRAN code, the evaluation of operators involving angular dimensions is partitioned into *nLoop* iterations. Within each iteration, the **ExchangeData** subroutine is called first, which gathers raw WF data needed by the calculation and scatters updated data blocks, and then, the actual calculations are performed. The integer variable *iLoop* denotes the current loop index (starting from 1). An additional iteration with the index "*nLoop + 1*" is necessary to finish scattering the last part of the updated wave function. Overlapping computation and communication is achieved by pre-fetching wave functions in the previous iteration.

3. Inter-process data exchange: One-sided MPI implement

In Sec. III C 2, the inter-process communication is based on a send/receive two-side pattern, as usually done in most of the reported MPI-based quantum dynamics code.

Consequently, two MPI processes (one as sender and another as receiver) have to be involved to complete the communication. In the second implementation of the parallel algorithm, we developed a much simpler architecture for communication, taking full advantage of the "one-side communication" feature supported by MPI-2[49] and its successors.

In a one-sided communication-based parallel implementation, the WF array belonging to each MPI process is exposed to all other processes through the **MPI_Win_Create** subroutine. Doing this makes the WF arrays accessible and modifiable by other processes through the **MPI_Get** and **MPI_Put** subroutines, logically equivalent to the read and write operations to local memory, respectively. The **initExchange** subroutine is no longer needed since the MPI reading and writing utilities are now encapsulated into the new **getRawData** and **putRstData** subroutines. In addition, the srBuf array used in the one-sided communication implement is now replaced with two separated arrays, i.e., **rawBuf(nRowp*kcol, 0:nProc-1)** and **rstBuf(nRowp*kcol, 0:nProc-1)**. The corresponding FORTRAN codes are presented in Appendix A 2.

Both the **MPI_Get** and **MPI_Put** functions are "non-blocking." This feature facilitates the overlap between the calculation and communication, similar to the combination of **MPI_ISend/MPI_IRecv** utilities in the two-side scheme. Other than its concision compared to the widely used two-sided pattern, one-sided communication has several advantages, such as reduced synchronization and communication time,[49] thus higher efficiency of parallelization as illustrated below.

4. Intra-process computation: OpenMP

The TDWP-5A9D program features a hybrid parallelization scheme, combining the above-mentioned MPI-based distributed-memory algorithm with a shared-memory OpenMP technology. The OpenMP portion of the program, mainly for parallel loops, was implemented in a traditional way, details of which may be found by interested readers in relevant textbooks.[50]

Here, we would like to analyze the necessity and emphasize the advantages of this hybrid scheme in the context of the current TDWP-5A9D program. First, with a pure MPI scheme, to take full advantage of the multi-core architecture of modern computers, more MPI processes need to be created, typically of the same number of local CPU cores. As seen in the above-mentioned algorithm, every MPI process has to communicate with all other processes. Generally, more MPI processes mean more significant overheads for network communication. In contrast, with a hybrid MPI/OpenMP scheme, the number of MPI processes is significantly reduced, resulting in higher efficiency.

Second, a better overlap between communication and computation can be achieved. With a pure MPI scheme, the number of columns distributed per iteration, *kcol*, cannot be too large even with a high network bandwidth, owing to the limited computation capability of a single CPU core. With a hybrid MPI/OpenMP scheme, due to the inherent very high parallel efficiency of OpenMP,

the value of *kcol* can be much larger than that in a pure MPI scheme. The value of *kcol* is optimal when the bandwidth of network communication is exhausted, i.e., a further increase in the data packet size leads to no better or even worse performance in communication.

IV. RESULTS AND DISCUSSIONS

We carried out calculations on one of our in-house computer clusters. The cluster comprises 30 shared-memory computer nodes, each with 20 Intel Xeon E5-2640 v4 processors (CPU cores) at 2.40 GHz and 256 gigabytes (GB) of main memory. The operating system running on the nodes is CentOS Linux release 7.2.1511 with the 3.10.0-327.el7.x86_64 kernel. Each node is equipped with one InfiniBand host channel adapter (HCA) supporting 4x Quad Data Rate (QDR) connections with a 40 GB/s speed. We have one 4x-QDR InfiniBand connection from each node to a central InfiniBand switch (Mellanox, MT26428). The TDWP-5A9D program was compiled at the -O2 optimization level with OpenMP support using the Intel Fortran compiler (version 15.0.2). The MPI utilities implemented as in Open MPI[51] (version 1.4.5) were adopted.

The most important issue is whether the parallel program can calculate the correct result. To answer this question, we have conducted reaction dynamics calculations on the H + NH_3 system and compared the reaction probabilities to the right results, where quite good agreement has been seen (not shown).

We performed timing runs for the H + NH_3 reaction system involving up to 50 steps of wave packet propagation. All the times are in wall clock time, obtained using the **MPI_WTime** subroutine. Two different test methods, named method-I and method-II, respectively, have been employed. Within method-I, which corresponds to the so-called "strong scaling,"[52] the number of MPI processes is increased gradually while the size of the simulation remains constant. In method-II, which is related to the case of "weak scaling," the number of MPI processes and the simulation size increase simultaneously, resulting in an essentially constant workload per MPI process.

Table I presents the parameters of the basis sets used in the time runs for these two test methods. The values of parameters used for method-I are close to the actual calculations.[43] Most parameter values used in the method-II case are identical to those in method-I, except for fewer *R* grids and basis functions. Note that the parameter values of method-II, as presented in Table I, correspond to the fundamental case with one MPI process; for other runs with more MPI processes, the *nR* parameter value was increased proportionally. The center of the initial Gaussian wave packet is located at 13.0 a_0 and the width of the packet is 0.32 a_0. For test purpose only, the ground state (00^+00) NH_3 reactant and $J_{tot} = 0$ was assumed, which was calculated in the same way as in our previous work.[43] The potential energy surface of NH_3 employed here was constructed by Li and Guo in 2014.[53]

Figure 3 shows the speedup and efficiency derived from method-I, i.e., with fixed parameter values and increased MPI process number. The speedup for a case of *n* MPI processes is defined as $SP_n = t_1/t_n$, where t_n is the running time taken on *n* MPI processes, each with 20 OpenMP threads. The speedup efficiency is defined as $EF_n = SP_n/n = (t_1/t_n)/n$.

TABLE I. Parameters of basis sets used in timing runs. Note that for the dimensions of *R* and R_3, different numbers of grid points are applied separately to the asymptotic and interaction regions. The parameter values for method-II correspond to the fundamental case run with one MPI process. The value of the *kcol* parameter is fixed to 40.

Parameters	nR	nR_3	nR_1	nR_2	$n\theta$	$n\beta_s$	$n\gamma_s$	$n\beta_Y$	$n\alpha_s$
Method-I	200/100	5/25	3	3	8	8	12	31	7
Method-II	20/10	5/25	3	3	8	8	12	31	7

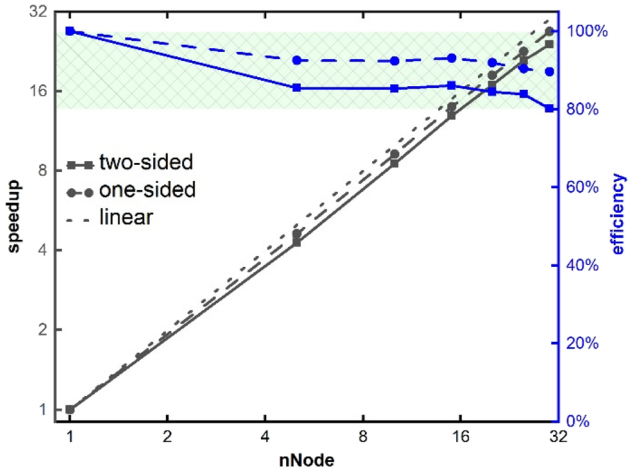

FIG. 3. Speedup and efficiency in method I with fixed parameter values and increased MPI process number (nNode). The dotted line shows the ideal linear speedup.

The tests were performed with the number of MPI processes ranging from 1 to 30. For both the two-sided (2S) and one-sided (1S) communication schemes implemented in this work, a linear scaling of speedup concerning the MPI process number (*nNode*) was observed. As a result, speedups of 24.04 and 26.88 were achieved with up to 30 MPI processes for the 2S and 1S schemes, respectively. For both the schemes, the speedup scaling deviates gradually from the ideal linear one ($SP_n = n$) with increased MPI processes. This reduction in the parallel performance can be attributed to the increased overhead of inter-process communications due to the fully connected nature of MPI processes involved and, thus, more data-exchanging operations.

For the 2S scheme, the speedup efficiency for 5 MPI processes is about 86%, which decreases slowly to about 80% for 30 MPI processes. The speedup efficiencies for the 1S scheme are higher than those for the 2S scheme, still about 90% for 30 MPI processes. An average enhancement of 7.4% in speedup efficiency is achieved by the 1S scheme compared to the 2S one, demonstrating the anticipated performance advantages of one-sided communication. Generally, we see excellent scalability (larger than 80% compared to the

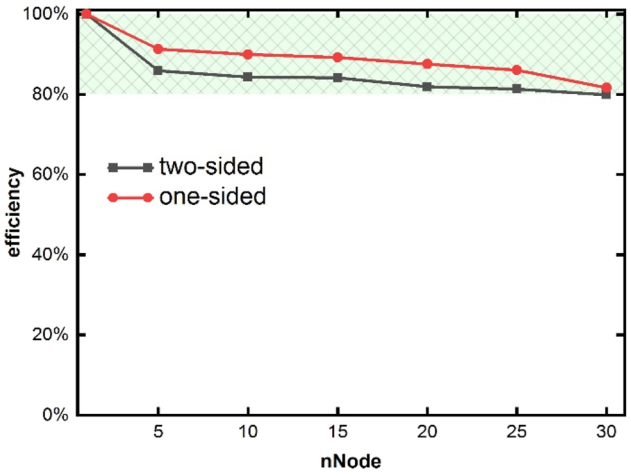

FIG. 4. Efficiency in method II with parameter values and MPI process number (nNode) increased proportionally.

ideal one) for both parallelization schemes, which can be obtained for up to 30 MPI processes (600 CPU cores).

Figure 4 shows the efficiency calculated according to method-II, i.e., parameter values and MPI process number increased proportionally, resulting in a constant workload per MPI process. Thus, different from method-I, the parallel efficiency for the case of n MPI processes is redefined as $EF_n' = t_1/t_n$. In an ideal situation, the efficiency value by this definition equals one, which means that one solves progressively larger problems in the same time as it takes to solve smaller ones using fewer computer nodes.

In practice, as shown in Fig. 4, the efficiency decreases along with increasing MPI processes for the 2S and 1S communication schemes. This trend of efficiency can be explained below. For more MPI processes, due to the increasing per-iteration simulation size ($nRot^* kcol^* nNode$), the latency of data exchange between MPI processes tends to increase, particularly for a network with limited bandwidth. In addition, with a simultaneously increased simulation size and process number, the impacts of intra-process computation–communication overlapping and inter-process load balancing on parallel performance may become more critical. Nonetheless, an excellent scalability (larger than 80%) concerning the total workload is maintained for up to 600 CPU cores. Similar to those shown in method-I, an average enhancement in the efficiency of about 5% was realized by the 1S scheme over the 2S one.

Finally, the influence of different *kcol* parameters was also investigated. The testing cases are based on simulation parameters adopted in method-I, which results in 9000 and 22 500 columns of the asymptotic and interaction wave functions, respectively. We measured the running time with 30 MPI processes for *kcol* values ranging from 20 to 300. It is found that the *kcol* parameter has a nontrivial impact on the parallelization performance, showing a relative difference in the clock time of roughly 15%–20%. Briefly speaking, a larger *kcol* value is preferred to exhaust the network bandwidth and achieve a better computation–communication overlap. An optimal *kcol* value was determined to be about 200, associated with the current simulation parameters and hardware configurations. For even larger *kcol* values, the measured clock time tends to be stable with a minor fluctuation of about 5%. Note that one may not pay too much attention to finding the optimal *kcol* value since the primary concern is usually whether the MPI application can help make what used to be an impossible or expensive calculation happen in a practical and cheap network.

V. CONCLUSIONS

This study introduces a parallel algorithm designed to conduct precise high-dimensional quantum dynamics simulations of poly atomic reactions, integrating both distributed- and shared-memory programming models. The distribution of the full wave function and potential energy matrices across MPI processes, employing bundled radial and angular dimensions, forms a foundational aspect of our approach. In addition, the implementation considerations include both two-sided and one-sided MPI communication schemes.

Our assessment of the algorithm's performance, using realistic parameters for a full-dimensional simulation of the H + NH_3 reaction, yielded promising results. Notably, the algorithm exhibited exceptional linear scalability, achieving efficiency levels surpassing 90% across 600 processors. Furthermore, our investigation demonstrated its remarkable extensibility to accommodate larger-sized simulations. The universality and concise structure of the algorithm enable seamless application to a broad spectrum of reaction systems. Building on our prior successes with reactions involving six and four atoms, the algorithm's adaptability is evident.

In summary, our work contributes a robust foundation for high-dimensional dynamics studies, laying the groundwork for future advancements in the field. The efficiency, scalability, and adaptability demonstrated by the parallel algorithm underscores its potential as a valuable tool in unraveling the complexities of quantum dynamics across various chemical reactions.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grant Nos. 22373002, 21873004, 22173097, and 22322305), the Natural Science Foundation of Anhui Province (Grant No. 1708085QB41), the Anhui Provincial Leading Talents Project, the Anhui Provincial Innovation Project of Returnees from Overseas Studies, the Innovation Program for Quantum Science and Technology (Grant No. 2021ZD0303305), and the Dalian Innovation Support Program (Grant No. 2021RD05). LYP acknowledges the financial support from the Ministry of Education, Singapore under its Academic Research Fund Tier 1 RG82/22.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Yong Zhou: Formal analysis (equal); Funding acquisition (equal); Investigation (equal); Methodology (equal); Software (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **Yunpeng Lu**: Formal analysis (equal); Funding acquisition (equal); Investigation (equal); Methodology (equal); Software (equal); Validation (equal); Writing – review & editing (equal). **Zhaojun Zhang**: Formal analysis (equal); Funding acquisition (equal); Investigation (equal); Software (equal); Validation (equal); Writing – review & editing (equal). **Dong H. Zhang**: Conceptualization (equal); Formal analysis (equal); Funding acquisition (equal); Project administration (equal); Resources (equal); Supervision (equal); Writing – review & editing (equal).

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding authors upon reasonable request.

APPENDIX A: FORTRAN ROUTINES ON MPI DATA EXCHANGE

1. Implement based on two-sided communication

```
subroutine ExchangeData(iLoop)
if(iLoop == 1) then
call packRawData(iLoop) ! prepare raw data for this loop(1).
call initExchange
call FinishExchange
call unpackRawData(iLoop) ! raw data for this loop(1) ready.
call packRawData(iLoop+1) ! prepare raw data for next loop(2).
call initExchange
else if(iLoop /= nLoop+1) then
call finishExchange
if(iLoop-2 >= 1) then
call unpackRstData(iLoop-2) ! store data updated@(iLoop-2)
end if
call packRstData(iLoop-1) ! data updated@(iLoop-1), ! pack & send@(iLoop), ready@(iLoop+1)
call unpackRawData(iLoop) ! raw data for this loop ready
if(iLoop+1 <= nLoop) then ! if any, call packRawData(iLoop+1) ! prepare raw data for next loop
end if
call initExchange
else
call finishExchange
call unpackRstData(iLoop-2) ! store data updated@(nLoop-1)
call packRstData(iLoop-1)
call initExchange
call finishExchange
call unpackRstData(iLoop-1) ! store data updated@(nLoop)
endif
end subroutine ExchangeData
```

2. Implement based on one-sided communication

```
subroutine ExchangeData(iLoop)
if(iLoop == 1) then
call getRawData(iLoop) ! get raw data for this loop(1)
call FinishExchange
call unpackRawData(iLoop) ! raw data for this loop(1) ready.
call getRawData(iLoop+1) ! get raw data for next loop(2).
else if(iLoop /= nLoop+1) then
call FinishExchange
call packRstData(iLoop-1)
call putRstData(iLoop-1) ! data updated@(iLoop-1)
call unpackRawData(iLoop) ! raw data for this loop ready
if(iLoop+1 <= nLoop) then ! if any
call getRawData(iLoop+1) ! get raw data for next loop
end if
else
call FinishExchange
call packRstData(iLoop-1)
call putRstData(iLoop-1) ! store data updated@(nLoop).
call FinishExchange
endif
end subroutine ExchangeData
```

APPENDIX B: USAGE OF THE TDWP-5A9D PROGRAM

The TDWP-5A9D program has been made freely available on the GitHub website (https://github.com/darrelzhang/TDWP-5A9D.git). For the parallelization strategy adopted by the TDWP-5A9D program, interested readers may refer to the source codes "*mpi_scatt_1.f*" and "*mpi_scatt_2.f*" for one-sided and two-sided MPI communication schemes, respectively.

An Intel Fortran compiler (version 2015 or later) and OpenMPI (version 1.4.5 or later) are required when compiling the program. It is essential to place all the files in the same directory. One should modify the variable "`MKL_PATH`" in the makefile and then run the "`make`" command to generate an executable file named *xynh2.exe*.

Before running the program, one specifies various parameters in the "*input*" file located in the same directory as the executable file. As needed, one can adjust the ranges and the number of basis functions/grid points for the nine degrees of freedom. It is necessary to utilize an appropriate absorbing potential to reduce the wave function's reflection at the boundary region. In order to obtain converged results, adjustments can be made to ensure a sufficient total propagation time and a small enough time step. The reaction probability is calculated at a specific position (*posfluxr*) within the exit region.

The program can be run with the command line "`mpirun -hostfile machinefile -n procs ./xynh2.exe`," where a file named "*machinefile*" contains the hostname of the parallel com-

puting nodes and the *procs* variable represents the total number of MPI processes. All the generated output files are placed in the *result* folder. Specifically, the time-dependent wave functions for the asymptotic and interaction regions are recorded in the "*asy.**" and "*int.**" files, respectively. The reaction probabilities are stored in the "*prob.eng*" and "*prob.eng.zz*" files.

REFERENCES

[1] R. D. Levine, *Molecular Reaction Dynamics* (Cambridge University Press, 2009).
[2] E. M. Goldfield and S. K. Gray, "Quantum dynamics of chemical reactions," in *Advances in Chemical Physics*, 1st ed., edited by S. A. Rice (Wiley, 2007), pp. 1–38.
[3] G. C. Schatz and A. Kuppermann, "Quantum mechanical reactive scattering for three-dimensional atom plus diatom systems. II. Accurate cross sections for H + H_2," J. Chem. Phys. **65**(11), 4668–4692 (1976).
[4] M. Mladenovic, M. Zhao, D. G. Truhlar, D. W. Schwenke, Y. Sun, and D. J. Kouri, "Converged quantum mechanical calculation of the product vibration-rotation state distribution of the hydrogen atom + para-hydrogen reaction," J. Phys. Chem. **92**(25), 7035–7038 (1988).
[5] A. Kuppermann and P. G. Hipes, "Three-dimensional quantum mechanical reactive scattering using symmetrized hyperspherical coordinates," J. Chem. Phys. **84**(10), 5962–5964 (1986).
[6] R. T. Pack and G. A. Parker, "Quantum reactive scattering in three dimensions using hyperspherical (APH) coordinates. Theory," J. Chem. Phys. **87**(7), 3888–3921 (1987).
[7] D. H. Zhang and J. Z. H. Zhang, "Full-dimensional time-dependent treatment for diatom–diatom reactions: The H_2 + OH reaction," J. Chem. Phys. **101**(2), 1146–1156 (1994).
[8] D. Neuhauser, "Fully quantal initial-state-selected reaction probabilities (J = 0) for a four-atom system: H_2(v = 0, 1, j = 0) + OH(v = 0, 1, j = 0) → H + H_2O," J. Chem. Phys. **100**(12), 9272–9275 (1994).
[9] D. H. Zhang and J. C. Light, "Quantum state-to-state reaction probabilities for the H + H_2O → H_2 + OH reaction in six dimensions," J. Chem. Phys. **105**(3), 1291–1294 (1996).
[10] B. Fu, X. Shan, D. H. Zhang, and D. C. Clary, "Recent advances in quantum scattering calculations on polyatomic bimolecular reactions," Chem. Soc. Rev. **46**(24), 7625–7649 (2017).
[11] B. Fu and D. H. Zhang, "Ab initio potential energy surfaces and quantum dynamics for polyatomic bimolecular reactions," J. Chem. Theory Comput. **14**(5), 2289–2303 (2018).
[12] X. Q. Zhang, Q. Cui, J. Z. H. Zhang, and K. L. Han, "Quantum dynamics study of H + NH_3 → H_2 + NH_2 reaction," J. Chem. Phys. **126**(23), 234304 (2007).
[13] M. Yang and J. C. Corchado, "Seven-dimensional quantum dynamics study of the H + NH_3 → H_2 + NH_2 reaction," J. Chem. Phys. **126**(21), 214312 (2007).
[14] S. Liu, J. Chen, Z. Zhang, and D. H. Zhang, "Communication: A six-dimensional state-to-state quantum dynamics study of the H + CH_4 → H_2 + CH_3 reaction (J = 0)," J. Chem. Phys. **138**(1), 011101 (2013).
[15] M. Yang, D. H. Zhang, and S.-Y. Lee, "A seven-dimensional quantum study of the H + CH_4 reaction," J. Chem. Phys. **117**(21), 9539–9542 (2002).
[16] W. Zhang, Y. Zhou, G. Wu, Y. Lu, H. Pan, B. Fu, Q. Shuai, L. Liu, S. Liu, L. Zhang, B. Jiang, D. Dai, S.-Y. Lee, Z. Xie, B. J. Braams, J. M. Bowman, M. A. Collins, D. H. Zhang, and X. Yang, "Depression of reactivity by the collision energy in the single barrier H + CD_4 → HD + CD_3 reaction," Proc. Natl. Acad. Sci. U. S. A. **107**(29), 12782–12785 (2010).
[17] R. Liu, H. Song, J. Qi, and M. Yang, "A ten-dimensional quantum dynamics model for the X + $YCAB_2$ reaction: Application to H + CH_4 reaction," J. Chem. Phys. **153**(22), 224119 (2020).
[18] M. Yang, "Full dimensional time-dependent quantum dynamics study of the H + NH_3 → H_2 + NH_2 reaction," J. Chem. Phys. **129**(6), 064315 (2008).
[19] U. Manthe, H.-D. Meyer, and L. S. Cederbaum, "Wave-packet dynamics within the multiconfiguration Hartree framework: General aspects and application to NOCl," J. Chem. Phys. **97**(5), 3199–3213 (1992).
[20] B. Zhao, Z. Sun, and H. Guo, "Calculation of state-to-state differential and integral cross sections for atom-diatom reactions with transition-state wave packets," J. Chem. Phys. **140**(23), 234110 (2014).
[21] B. Zhao, Z. Sun, and H. Guo, "State-to-state mode specificity: Energy sequestration and flow gated by transition state," J. Am. Chem. Soc. **137**(50), 15964–15970 (2015).
[22] B. Zhao and H. Guo, "Modulations of transition-state control of state-to-state dynamics in the F + H_2O → HF + OH reaction," J. Phys. Chem. Lett. **6**(4), 676–680 (2015).
[23] H. Wang, "Multilayer multiconfiguration time-dependent Hartree theory," J. Phys. Chem. A **119**(29), 7951–7965 (2015).
[24] H.-D. Meyer, U. Manthe, and L. S. Cederbaum, "The multi-configurational time-dependent Hartree approach," Chem. Phys. Lett. **165**(1), 73–78 (1990).
[25] R. Welsch and U. Manthe, "Reaction dynamics with the multi-layer multi-configurational time-dependent Hartree approach: H + CH_4 → H_2 + CH_3 rate constants for different potentials," J. Chem. Phys. **137**(24), 244106 (2012).
[26] F. Huarte-Larrañaga and U. Manthe, "Full dimensional quantum calculations of the CH_4 + H → CH_3 + H_2 reaction rate," J. Chem. Phys. **113**(13), 5115–5118 (2000).
[27] T. Wu, H.-J. Werner, and U. Manthe, "First-principles theory for the H + CH_4 → H_2 + CH_3 reaction," Science **306**(5705), 2227–2229 (2004).
[28] R. Van Harrevelt, G. Nyman, and U. Manthe, "Accurate quantum calculations of the reaction rates for H/D + CH_4," J. Chem. Phys. **126**(8), 084303 (2007).
[29] G. Schiffel and U. Manthe, "Communications: A rigorous transition state based approach to state-specific reaction dynamics: Full-dimensional calculations for H + CH_4 → H_2 + CH_3," J. Chem. Phys. **132**(19), 191101 (2010).
[30] R. Welsch and U. Manthe, "Communication: Ro-vibrational control of chemical reactivity in H + CH_4 → H_2 + CH_3: Full-dimensional quantum dynamics calculations and a sudden model," J. Chem. Phys. **141**(5), 051102 (2014).
[31] R. Welsch and U. Manthe, "Full-dimensional and reduced-dimensional calculations of initial state-selected reaction probabilities studying the H + CH_4 → H_2 + CH_3 reaction on a neural network PES," J. Chem. Phys. **142**(6), 064309 (2015).
[32] J. Qi, H. Song, M. Yang, J. Palma, U. Manthe, and H. Guo, "Communication: Mode specific quantum dynamics of the F + CHD_3 → HF + CD_3 reaction," J. Chem. Phys. **144**(17), 171101 (2016).
[33] Z. Zhang, F. Gatti, and D. H. Zhang, "Full-dimensional quantum mechanical calculations of the reaction probability of the H + CH_4 reaction based on a mixed Jacobi and Radau description," J. Chem. Phys. **152**(20), 201101 (2020).
[34] E. M. Goldfield and S. K. Gray, "Mapping Coriolis-coupled quantum dynamics onto parallel computer architectures," Comput. Phys. Commun. **98**(1–2), 1–14 (1996).
[35] M. T. Cvitas and S. C. Althorpe, "Parallelizable split-operator propagator for treating Coriolis-coupled quantum dynamics," Comput. Phys. Commun. **177**(4), 357–361 (2007).
[36] P. Eggert, A. Viel, and C. Leforestier, "Parallel implementation of a pseudo-spectral calculation of molecular energy levels: Application to the water dimer $(H_2O)_2$," Comput. Phys. Commun. **128**(1–2), 315–325 (2000).
[37] E. M. Goldfield, "Parallel strategies for four-atom quantum dynamics calculations," Comput. Phys. Commun. **128**(1–2), 178–189 (2000).
[38] A. J. H. M. Meijer, "Time-dependent wave packet calculations on parallel computers: A new and efficient algorithm for evaluating HΨ," Comput. Phys. Commun. **141**(3), 330–341 (2001).
[39] S. Borowski, S. Thiel, T. Klüner, H.-J. Freund, R. Tisma, and H. Lederer, "High-dimensional quantum dynamics of molecules on surfaces: A massively parallel implementation," Comput. Phys. Commun. **143**(2), 162–173 (2002).
[40] D. M. Medvedev, E. M. Goldfield, and S. K. Gray, "An OpenMP/MPI approach to the parallelization of iterative four-atom quantum mechanics," Comput. Phys. Commun. **166**(2), 94–108 (2005).
[41] M. Brill, O. Vendrell, F. Gatti, and H.-D. Meyer, "Shared memory parallelization of the multiconfiguration time-dependent Hartree method and application to the dynamics and spectroscopy of the protonated water-dimer," in *High Performance Computing in Science and Engineering'07*, edited by W. E. Nagel, D. Kröner, and M. Resch (Springer, Berlin, Heidelberg, 2008), pp. 141–155.

[42]M. Brill, O. Vendrell, and H.-D. Meyer, "Distributed memory parallelization of the multiconfiguration time-dependent Hartree method," in *High Performance Computing in Science and Engineering'09*, edited by W. E. Nagel, D. B. Kröner, and M. M. Resch (Springer, Berlin, Heidelberg, 2010), pp. 147–163.
[43]Z. Zhang, F. Gatti, and D. H. Zhang, "Full dimensional quantum mechanical calculations of the reaction probability of the H + NH_3 collision based on a mixed Jacobi and Radau description," J. Chem. Phys. **150**(20), 204301 (2019).
[44]B. Fu, Y. Zhou, and D. H. Zhang, "Shape resonance in the H + D_2O → D + HOD reaction: A full-dimensional quantum dynamics study," Chem. Sci. **3**(1), 270–274 (2012).
[45]Z. Zhao, J. Chen, Z. Zhang, D. H. Zhang, D. Lauvergnat, and F. Gatti, "Full-dimensional vibrational calculations of five-atom molecules using a combination of Radau and Jacobi coordinates: Applications to methane and fluoromethane," J. Chem. Phys. **144**(20), 204302 (2016).
[46]Z. Zhao, J. Chen, Z. Zhang, D. H. Zhang, X.-G. Wang, T. Carrington, and F. Gatti, "Computing energy levels of CH_4, CHD_3, CH_3D, and CH_3F with a direct product basis and coordinates based on the methyl subsystem," J. Chem. Phys. **148**(7), 074113 (2018).
[47]M. D. Feit and J. A. Fleck, Jr., "Solution of the Schrödinger equation by a spectral method II: Vibrational energy levels of triatomic molecules," J. Chem. Phys. **78**(1), 301–308 (1983).
[48]B. Satarić, V. Slavnić, A. Belić, A. Balaž, P. Muruganandam, and S. K. Adhikari, "Hybrid OpenMP/MPI programs for solving the time-dependent Gross–Pitaevskii equation in a fully anisotropic trap," Comput. Phys. Commun. **200**, 411–417 (2016).
[49]Message Passing Interface Forum, MPI: A Message-Passing Interface Standard Version 2.0, 1997.
[50]B. Chapman, G. Jost, and R. van der Pas, *Using OpenMP: Portable Shared Memory Parallel Programming* (MIT Press, Cambridge, MA, 2008).
[51]R. L. Graham, T. S. Woodall, and J. M. Squyres, "Open MPI: A flexible high performance MPI," in *Parallel Processing and Applied Mathematics*, edited by R. Wyrzykowski, J. Dongarra, N. Meyer, and J. Waśniewski (Springer, Berlin, Heidelberg, 2006), pp. 228–239.
[52]*Performance Tuning of Scientific Applications*, edited by D. H. Bailey, R. F. Lucas, and S. Williams (CRC Press, 2010).
[53]J. Li and H. Guo, "A nine-dimensional global potential energy surface for NH_4 (X^2A_1) and kinetics studies on the H + NH_3 ↔ H_2 + NH_2 reaction," Phys. Chem. Chem. Phys. **16**(14), 6753–6763 (2014).