JCTC Journal of Chemical Theory and Computation

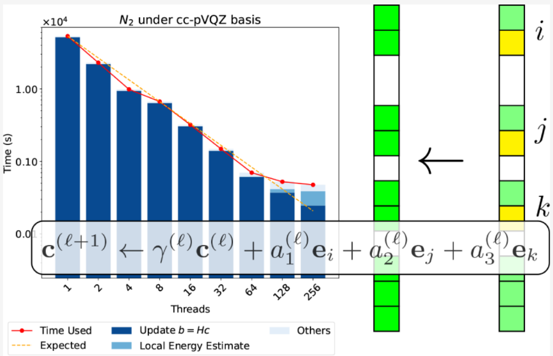
pubs.acs.org/JCTC Article

Parallel Multicoordinate Descent Methods for Full Configuration Interaction

Yuejia Zhang, Weiguo Gao,* and Yingzhou Li*

Cite This: J. Chem. Theory Comput. 2025, 21, 2325−2337

ABSTRACT: We develop a multithreaded parallel coordinate descent full configuration interaction algorithm (mCDFCI) for the electronic structure ground-state calculation in the configuration interaction framework. The FCI problem is reformulated as an unconstrained minimization problem and tackled by a modified block coordinate descent method with a deterministic compression strategy. mCDFCI is designed to prioritize determinants based on their importance, with block updates enabling efficient parallelization on shared-memory, multicore computing infrastructure. We demonstrate the efficiency of the algorithm by computing an accurate benchmark energy for the chromium dimer in the Ahlrichs SV basis (48e, 42o), which explicitly includes 2.07×10^9 variational determinants. We also provide the binding curve of the nitrogen dimer under the cc-pVQZ basis set (14e, 110o). Benchmarks show up to 79.3% parallel efficiency on 128 cores.



1. INTRODUCTION

Understanding the chemical properties of molecules relies on solving the many-body time-independent electronic Schrödinger equation. However, traditional methods, such as density functional theory (DFT) or coupled-cluster with single, double, and perturbative triple excitations (CCSD(T)), often struggle to accurately describe the electronic structure of strongly correlated systems. This limitation is particularly evident in molecules with transition metals or those in nonequilibrium geometries.

Full configuration interaction (FCI) provides a numerically exact solution under a predefined basis set by describing the wave function as a superposition of all possible Slater determinants. However, FCI methods scale exponentially with the number of orbitals and electrons, leading to the curse of dimensionality. To overcome this challenge and apply FCI methods to large systems, it becomes necessary to compress the wave function. This can be achieved by employing different wave function ansatze, such as the matrix product state (MPS) in the density matrix renormalization group (DMRG) method,[1−5] or by representing the wave function as a population of random particles, as in the full configuration interaction quantum Monte Carlo (FCIQMC) method.[6−10] Another approach involves selecting important Slater determinants, guided by the extensive sparsity of the FCI wave function.[11] The method we describe in this paper falls into this category, which is known as the selected CI method.

A variety of selected CI methods have been developed, starting from the earliest work in 1973 known as Configuration Interaction using a Perturbative Selection done Iteratively (CIPSI),[12] to recent advancements including Adaptive Sampling CI (ASCI),[13] Heat-bath CI (HCI),[14] Semistochastic Heat-bath CI (SHCI),[15,16] Coordinate Descent FCI (CDFCI),[17] Fast Randomized Iteration method for FCI (FCI-FRI),[18] Reinforcement Learning CI (RLCI),[19] and others. These methods share the common iterative approach of expanding a primary configuration space, filtering determinants based on some importance estimate, and computing the leading eigenpair to obtain an enhanced approximation of the wave function until convergence is achieved. Typically, Epstein−Nesbet second-order perturbation theory is further employed in the secondary space to account for the remaining correlation that is not captured by the variational SCI treatment. Selected CI variants significantly reduce the computational cost of FCI by diminishing the dimension of the primary SCI space compared to the N-electron Hilbert space, while they differ by having distinct selection principles and implementations.

This paper extends the coordinate descent FCI (CDFCI)[17] method previously proposed by one of our authors and his

Received: November 11, 2024
Revised: February 11, 2025
Accepted: February 12, 2025
Published: February 28, 2025

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

collaborators, which provides selection rules from an optimization perspective. Initially, it transforms the FCI eigenvalue problem into an unconstrained minimization problem, with local minima corresponding to the ground state of the system. Next, it employs the coordinate descent method for the following advantages: (i) The gradient of the objective function provides a natural determinant selection rule, adding important determinants into the variational space until it reaches the memory limit; (ii) The special structure of the problem allows us to perform an exact line search, accelerating the energy convergence; (iii) in each iteration, updating only one coordinate of the optimization vector involves only one column of the Hamiltonian matrix, avoiding operations with the entire Hamiltonian matrix, thus reducing the computation cost associated with unappreciative determinants. The CDFCI method obtains the ground-state energy and wave function without explicitly extracting the Hamiltonian submatrix for direct diagonalization. This makes immense room for the storage of the wave function, making it possible for larger systems and more accurate approximations. The effective determinant selection rule and the low storage cost are the main reasons why CDFCI becomes a competitive FCI solver.

Although CDFCI demonstrates accelerated performance in experiments, its parallelization capability is restricted by the inherent sequential nature of the method. In this paper, we present a novel algorithm to address the minimization problem, which extends the update of one coordinate to multiple coordinates per iteration. To achieve this, the algorithm introduces an additional search dimension to enable the exact line search. This extension not only accelerates convergence but also opens up new possibilities for parallelization. Benefiting from fully parallelizable coordinate updates, our new algorithm achieves an accuracy of 10^{-5} Ha for C_2 and N_2 using the cc-pVDZ basis in 10 and 30 min respectively, nearly 20 times faster than the single-threaded version reported in the original CDFCI work. When compared to the multi-threaded version of the original CDFCI that supports parallel hash table updates, our algorithm delivers a 3.0 × speedup. Additionally, it computes the ground state of all-electron Cr_2 with Ahlrichs SV basis in 5.8 days, matching the accuracy of the original CDFCI, which previously required one month for the same task.

In the rest of this paper, we present the algorithm in Section 2 and discuss the implementation details in Section 3. In Section 4, we demonstrate the accuracy and the parallel efficiency of our method by applying it to various molecules, including C_2, N_2 and Cr_2. The binding curve of N_2 under the cc-pVQZ basis is also characterized. Finally, we conclude and look ahead to future work in Section 5.

2. METHODOLOGY AND APPROACH

In this section, we begin by reviewing the reformulation of the FCI eigenvalue problem as a nonconvex optimization problem.[17,20] Following this, we describe in detail the multicoordinate descent methods to address the FCI problem.

2.1. Problem Formulation. With a complete set of one-electron spin−orbitals $\{\chi_p\}$, the many-body Hamiltonian operator can be expressed using second quantization as follows:

$$\hat{H} = \sum_{p,q} t_{pq}\hat{a}_p^\dagger \hat{a}_q + \frac{1}{2}\sum_{p,q,r,s} v_{pqrs}\hat{a}_p^\dagger \hat{a}_q^\dagger \hat{a}_s \hat{a}_r \tag{1}$$

where $\hat{a}_p^\dagger$ and $\hat{a}_p$ represent the creation and annihilation operators, respectively, for spin−orbital p. The coefficients t_{pq} and v_{pqrs} correspond to one- and two-electron integrals. The ground-state energy of $\hat{H}$ is determined by solving the time-independent Schrödinger equation:

$$\hat{H}|\Phi_0\rangle = E_0|\Phi_0\rangle \tag{2}$$

where E_0 represents the ground-state energy, i.e., the lowest eigenvalue of $\hat{H}$, and $|\Phi_0\rangle$ is the associated ground-state wave function. We assume, without loss of generality, that E_0 is negative and nondegenerate, meaning the eigenvalues of $\hat{H}$ are ordered as $E_0 < E_1 \le E_2 \le \cdots$.

The full configuration interaction (FCI) method starts from a truncated finite spin−orbital subset $\{\chi_p\}_{p=1}^{n_{\text{orb}}}$, where n_{orb} denotes the number of orbitals, and this subset is usually obtained from a Hartree−Fock procedure. The wave function is approximated as a linear combination of Slater determinants,

$$|\Phi_0\rangle = \sum_{i=1}^{N_{\text{FCI}}} c_i|D_i\rangle = \sum_{i=1}^{N_{\text{FCI}}} c_i|\chi_{i_1}\chi_{i_2}\cdots\chi_{i_n}\rangle \tag{3}$$

with $1 \le i_1 < i_2 < \cdots < i_n \le n_{\text{orb}}$. Here, $n = n_{\text{elec}}$ denotes the number of electrons in the system, and N_{FCI} is the dimension of FCI variational space, which represents the total number of configurations, and is of order $O\left(\binom{n_{\text{orb}}}{n_{\text{elec}}}\right)$.

Following this, the Schrödinger equation (2) is discretized into a matrix eigenvalue problem

$$H\mathbf{c} = E_0\mathbf{c} \tag{4}$$

where H is a real symmetric $N_{\text{FCI}} \times N_{\text{FCI}}$ matrix, with entry $H_{i,j} = \langle D_i|\hat{H}|D_j\rangle$, and $\mathbf{c}$ is a vector of dimension N_{FCI}, with entry c_i. The problem (eq 4) is known as the FCI eigenvalue problem.

The second-quantized Hamiltonian operator defined in (eq 1) signifies that $H_{i,j}$ is zero if transforming $|D_i\rangle$ into $|D_j\rangle$ involves altering more than two occupied spin−orbitals. Consequently, H has $O(n_{\text{elec}}^2 n_{\text{orb}}^2)$ nonzero entries per row, which is much smaller than N_{FCI} in terms of order, thus leading to an extreme sparsity structure in the matrix H.

Selected CI methods[12−15] target a submatrix of H and use its lowest eigenpair as an estimate for the FCI eigenvalue problem. Now, instead of performing direct diagonalization on the submatrix of H, we consider the following optimization problem

$$\min_{c\in\mathbb{R}^{N_{\text{FCI}}}} f(\mathbf{c}) = \min_{\mathbf{c}} \left\| H + \mathbf{c}\mathbf{c}^{\text{T}} \right\|_{\text{F}}^2 \tag{5}$$

with the same Hamiltonian matrix H as defined above. The gradient of the objective function is

$$\nabla f(\mathbf{c}) = 4H\mathbf{c} + 4(\mathbf{c}^{\text{T}}\mathbf{c})\mathbf{c} \tag{6}$$

It is shown that[20] $\pm\sqrt{-E_0}\,\mathbf{v}_0$ are the only two local minimizers, where $\mathbf{v}_0$ is the normalized eigenvector corresponding to the smallest eigenvalue $E_0 < 0$. Suppose we have an estimate $\hat{\mathbf{c}}$ of the solution $\mathbf{c}^*$ of problem (eq 5), by simply normalizing $\hat{\mathbf{c}}$, we can obtain an approximate solution of the FCI eigenvalue problem.

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

In this work, we follow the idea of coordinate descent, extending one coordinate update per iteration to multiple coordinates, and add an additional scaling factor to enable exact line search, which is undoubtedly more beneficial to convergence. We also follow the compression strategy in CDFCI, which improves the efficiency of the algorithm and controls the memory footprint so that CDFCI can converge in a subspace. Below, we describe our algorithm in detail.

2.2. Algorithm. The multicoordinate descent method is an iterative optimization technique similar to the original coordinate descent method. However, instead of modifying a single coordinate of the optimization variable $\mathbf{c}$ in each iteration, we update k coordinates simultaneously and introduce a scaling factor γ.

Let us denote $\mathbf{c}^{(l)}$ as the vector $\mathbf{c}$ at the current iteration l. We also define $\mathcal{I}^{(l)} = \{i_1^{(l)}, ..., i_k^{(l)}\} \subset \{1, ..., N_{\mathrm{FCI}}\}$ as the set of k distinct coordinates chosen for the update in this iteration.

The update rule for each component of $\mathbf{c}$ is given by

$$c_i^{(l+1)} \leftarrow \begin{cases} \gamma^{(l)} c_i^{(l)} + a_i^{(l)}, & \text{if } i \in \mathcal{I}^{(l)}, \\ \gamma^{(l)} c_i^{(l)}, & \text{otherwise} \end{cases} \tag{7}$$

where $\gamma^{(l)}$ is a scaling factor, and $a_i^{(l)}$ is the corresponding step size. Each entry $c_i^{(l)}$ is initially scaled by $\gamma^{(l)}$. For the specific coordinates i included in the set $\mathcal{I}^{(l)}$, $c_i^{(l)}$ is then further updated by adding a step size to the current value.

In matrix form, the update rule can be expressed as

$$\mathbf{c}^{(l+1)} \leftarrow \gamma^{(l)} \mathbf{c}^{(l)} + \mathcal{E}_{\mathcal{I}^{(l)}} \mathbf{a}^{(l)} \tag{8}$$

where $\mathcal{E}_{\mathcal{I}} \in \mathbb{R}^{N_{\mathrm{FCI}} \times k}$ is defined as $\mathcal{E}_{\mathcal{I}} = [e_{i_1}, ..., e_{i_k}]$, with $\mathcal{I} = \{i_1, ..., i_k\}$ and $1 \leq i_j \leq N_{\mathrm{FCI}}$. In other words, $\mathcal{E}_{\mathcal{I}}$ consists of columns from the identity matrix corresponding to the selected coordinates. Essentially, it projects the elements of $\mathbf{a}^{(l)} \in \mathbb{R}^k$ into the higher-dimensional space of $\mathbf{c}^{(l)} \in \mathbb{R}^{N_{\mathrm{FCI}}}$, inserting zeros for the coordinates not included in $\mathcal{I}^{(l)}$.

In Section 2.2.1, we discuss how we select the k coordinates in each iteration. Next, in Section 2.2.2, we explain how we choose the scaling scalar $\gamma^{(l)}$ and the stepsize vector $\mathbf{a}^{(l)}$, so that they give the best descent value in the objective function. Each local update actually provides an energy estimate for the global problem.

Following the algorithmic design in CDFCI,[20] we store an additional vector $\mathbf{b}$ in memory, which approximates $H\mathbf{c}$ with differences only in the coordinates that are compressed. This will facilitate both picking coordinates and choosing the stepsize. In Section 2.2.3, we show how we update $\mathbf{b}$ and $\mathbf{c}$ as well as other quantities after each iteration, where we insert a compression strategy that keeps the memory footprint affordable. Finally, in Section 2.2.4, we provide the complete pseudocode, along with a discussion of possible choices for initialization and stopping criteria.

2.2.1. Coordinate Selection. The selection rule of coordinates follows the largest gradients rule: the k coordinates with the largest absolute value of (eq 6) are considered. In other words, choose k coordinates, in which

$$i_j = \underset{i \neq i_1, ..., i_{j-1}}{\arg\max} |[\nabla f(\mathbf{c})]_i|, \; j = 1, ..., k \tag{9}$$

where $[\nabla f(\mathbf{c})]_i$ denotes the i-th coordinate of $\nabla f(\mathbf{c}) = 4H\mathbf{c} + 4(\mathbf{c}^{\mathrm{T}}\mathbf{c})\mathbf{c}$.

Since in the algorithm design, the vector $\mathbf{b}$ is stored as a compressed representation of $H\mathbf{c}$, at iteration l, the approximated gradient of the objective function can be easily obtained by a linear combination of $\mathbf{b}^{(l)}$ and $\mathbf{c}^{(l)}$. In addition, we consider only the determinant set $\mathcal{I}_H(\mathcal{I}^{(l)})$ which is defined as

$$\mathcal{I}_H(\mathcal{I}^{(l)}) := \left\{1 \leq i \leq N_{\mathrm{FCI}}: \exists\, j \in \mathcal{I}^{(l)} \text{ such that } H_{i,j} \neq 0\right\}$$

where $\mathcal{I}^{(l)}$ denotes the set of coordinates selected during the l-th iteration.

Therefore, the actual coordinate selection rule at iteration l is

$$i_j^{(l)} = \underset{\substack{i \in \mathcal{I}_H(\mathcal{I}^{(l-1)}) \\ i \neq i_1^{(l)}, ..., i_{j-1}^{(l)}}}{\arg\max} |4b_i^{(l)} + 4((\mathbf{c}^{(l)})^{\mathrm{T}}\mathbf{c}^{(l)})c_i^{(l)}|, \quad j = 1, ..., k \tag{10}$$

Our coordinate selection method is built on the concept of an active space, which is a carefully chosen subset of the full configuration space. This active space includes only the most important determinants, that are expected to contribute the most to the solution. To construct this space efficiently, we limit our consideration to determinants that are connected to the current iterate by the creation or annihilation of up to two electrons. This restriction helps to avoid including determinants that are far removed in configuration space, which would contribute minimally to the final solution. This approach enhances computational efficiency by focusing resources on the most relevant determinants, ensuring that the optimization process is both effective and efficient.

2.2.2. Optimal Scaling and Stepsize Selection. Once we have selected $\mathcal{I}^{(l)}$, the k coordinates to update, we minimize the objective function with respect to the scaling scalar $\gamma^{(l)}$ and the stepsize vector $\mathbf{a}^{(l)}$:

$$f(\mathbf{c}^{(l+1)}) = \left\| H + \mathbf{c}^{(l+1)} (\mathbf{c}^{(l+1)})^{\mathrm{T}} \right\|_{\mathrm{F}}^2 \tag{11}$$

where $\mathbf{c}^{(l+1)} = \gamma^{(l)}\mathbf{c}^{(l)} + \mathcal{E}_{\mathcal{I}^{(l)}}\mathbf{a}^{(l)}$. In the following, we describe how we perform an exact line search in a $(k+1)$-dimensional subspace so that $\gamma^{(l)}$ and $\mathbf{a}^{(l)}$ are both globally optimal.

Let $\mathbf{y}^{(l)} = \mathbf{c}^{(l)} - \mathcal{E}_{\mathcal{I}^{(l)}}(\mathcal{E}_{\mathcal{I}^{(l)}})^{\mathrm{T}}\mathbf{c}^{(l)}$, which sets the entries in coordinate set $\mathcal{I}^{(l)}$ to zeros. Define

$$\tilde{\mathbf{y}}^{(l)} = \begin{cases} \mathbf{y}^{(l)} / \left\|\mathbf{y}^{(l)}\right\|, & \text{if } \left\|\mathbf{y}^{(l)}\right\| \neq 0, \\ 0, & \text{otherwise} \end{cases}$$

so that we always have $\mathbf{y}^{(l)} = \tilde{\mathbf{y}}^{(l)} \left\|\mathbf{y}^{(l)}\right\|$ regardless of whether $\left\|\mathbf{y}^{(l)}\right\| = 0$.

The update of $\mathbf{c}$ given in (eq 8) can be rewritten as

$$\mathbf{c}^{(l+1)} = \gamma^{(l)}\mathbf{c}^{(l)} + \mathcal{E}_{\mathcal{I}^{(l)}}\mathbf{a}^{(l)} = Q^{(l)}\mathbf{z}^{(l)} \tag{12}$$

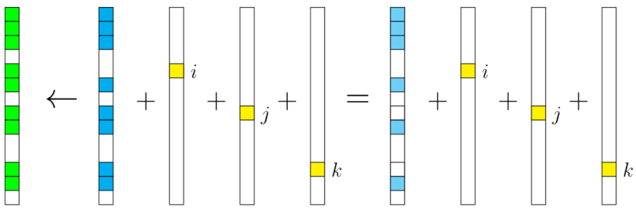
with $Q^{(l)}$ and $\mathbf{z}^{(l)}$ defined as follows:

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

$$Q^{(l)} := [\tilde{\mathbf{y}}^{(l)} \quad \mathcal{E}_{\mathcal{I}^{(l)}}] \in \mathbb{R}^{N_{\mathrm{FCI}} \times (k+1)} \tag{13}$$

$$\mathbf{z}^{(l)} := \begin{bmatrix} \gamma^{(l)} \|\mathbf{y}^{(l)}\| \\ \gamma^{(l)} (\mathcal{E}_{\mathcal{I}^{(l)}})^{\mathrm{T}} \mathbf{c}^{(l)} + \mathbf{a}^{(l)} \end{bmatrix} \tag{14}$$

Figure 1 demonstrates how the vector $\mathbf{c}$ is updated in each iteration.



$$\mathbf{c}^{(\ell+1)} \leftarrow \gamma^{(\ell)}\mathbf{c}^{(\ell)} + a_1^{(\ell)}\mathbf{e}_i + a_2^{(\ell)}\mathbf{e}_j + a_3^{(\ell)}\mathbf{e}_k = z_1^{(\ell)}\tilde{\mathbf{y}}^{(\ell)} + z_2^{(\ell)}\mathbf{e}_i + z_3^{(\ell)}\mathbf{e}_j + z_4^{(\ell)}\mathbf{e}_k$$

Figure 1. Update of vector $\mathbf{c}$ when $k = 3$. After three coordinates i, j, k are picked (in yellow), these coordinates of $\mathbf{c}^{(l)}$ are zeroed out and the resulting $\mathbf{y}^{(l)}$ is normalized to $\tilde{\mathbf{y}}^{(l)}$ (in light blue). The coefficients $a_i^{(l)}$ and $z_i^{(l)}$ represent the i-th element of vectors $\mathbf{a}^{(l)}$ and $\mathbf{z}^{(l)}$ respectively.

If $\|\mathbf{y}^{(l)}\| \neq 0$, $Q^{(l)}$ is an orthogonal matrix and an orthogonal basis for the $(k+1)$-dimensional search space. If $\|\mathbf{y}^{(l)}\| = 0$, $(Q^{(l)})^{\mathrm{T}} Q^{(l)} = \begin{bmatrix} 0 & \\ & I_k \end{bmatrix}$, which means the last k columns of $Q^{(l)}$ form an orthogonal basis for the k-dimensional search space.

Now consider splitting the optimization problem (11) into two parts as follows:

$$\begin{aligned} f(\mathbf{c}^{(l+1)}) &= \|H + \mathbf{c}^{(l+1)}(\mathbf{c}^{(l+1)})^{\mathrm{T}}\|_{\mathrm{F}}^2 \\ &= \|(Q^{(l)}(Q^{(l)})^{\mathrm{T}})(H + \mathbf{c}^{(l+1)}(\mathbf{c}^{(l+1)})^{\mathrm{T}})\|_{\mathrm{F}}^2 \\ &+ \|(I - Q^{(l)}(Q^{(l)})^{\mathrm{T}})(H + \mathbf{c}^{(l+1)}(\mathbf{c}^{(l+1)})^{\mathrm{T}})\|_{\mathrm{F}}^2 \\ &= \|(Q^{(l)})^{\mathrm{T}}(H + \mathbf{c}^{(l+1)}(\mathbf{c}^{(l+1)})^{\mathrm{T}})Q^{(l)}\|_{\mathrm{F}}^2 \\ &+ \|(I - Q^{(l)}(Q^{(l)})^{\mathrm{T}})H\|_{\mathrm{F}}^2 \\ &= \|(Q^{(l)})^{\mathrm{T}} H Q^{(l)} + \mathbf{z}^{(l)}(\mathbf{z}^{(l)})^{\mathrm{T}}\|_{\mathrm{F}}^2 + \text{constant} \end{aligned} \tag{15}$$

The third equality follows from the fact that $(I - Q^{(l)}(Q^{(l)})^{\mathrm{T}})\mathbf{c}^{(l+1)} = 0$, and the last equality is derived from eq 12. In addition, the term $\|(I - Q^{(l)}(Q^{(l)})^{\mathrm{T}})H\|_{\mathrm{F}}^2$ is considered as a constant because it does not depend on $\mathbf{z}^{(l)}$, and therefore neither on $\gamma^{(l)}$ nor $\mathbf{a}^{(l)}$.

We choose $\mathbf{z}^{(l)}$ to be the optimal solution to the local minimization problem

$$\begin{aligned} \mathbf{z}^{(l)} &= \arg\min_{\mathbf{z} \in \mathbb{R}^{k+1}} f(\mathbf{z}) \\ &= \arg\min_{\mathbf{z} \in \mathbb{R}^{k+1}} \|(Q^{(l)})^{\mathrm{T}} H Q^{(l)} + \mathbf{z}\mathbf{z}^{\mathrm{T}}\|_{\mathrm{F}}^2 \end{aligned} \tag{16}$$

The minimum is attained when[20]

$$\mathbf{z}^{(l)} = \pm\sqrt{-\lambda^{(l)}}\,\mathbf{v}_{\lambda^{(l)}} \tag{17}$$

where $\lambda^{(l)} := \lambda_{min}((Q^{(l)})^{\mathrm{T}} H Q^{(l)})$ is the minimal eigenvalue of matrix $(Q^{(l)})^{\mathrm{T}} H Q^{(l)}$ and $\mathbf{v}_{\lambda^{(l)}}$ is the corresponding eigenvector. The eigenvalue $\lambda^{(l)}$ is guaranteed to be negative as long as $(\mathbf{c}^{(1)})^{\mathrm{T}} H\, \mathbf{c}^{(1)} < 0$ (see Appendix A for proof and Section 2.2.4 for the discussion of initialization).

The problem (eq 16) consists of two parts: assembling the $(Q^{(l)})^{\mathrm{T}} H Q^{(l)}$ matrix and calculating its leading eigenpair. To start, we see that,

$$\begin{aligned} (Q^{(l)})^{\mathrm{T}} H Q^{(l)} &= \begin{bmatrix} (\tilde{\mathbf{y}}^{(l)})^{\mathrm{T}} \\ \mathcal{E}_{\mathcal{I}^{(l)}}^{\mathrm{T}} \end{bmatrix} H [\tilde{\mathbf{y}}^{(l)} \quad \mathcal{E}_{\mathcal{I}^{(l)}}] \\ &= \begin{bmatrix} (\tilde{\mathbf{y}}^{(l)})^{\mathrm{T}} H \tilde{\mathbf{y}}^{(l)} & (\tilde{\mathbf{y}}^{(l)})^{\mathrm{T}} H \mathcal{E}_{\mathcal{I}^{(l)}} \\ \mathcal{E}_{\mathcal{I}^{(l)}}^{\mathrm{T}} H \tilde{\mathbf{y}}^{(l)} & \mathcal{E}_{\mathcal{I}^{(l)}}^{\mathrm{T}} H \mathcal{E}_{\mathcal{I}^{(l)}} \end{bmatrix} \end{aligned} \tag{18}$$

If $\|\mathbf{y}^{(l)}\| \neq 0$, each block is computed as described below.

1. $(\tilde{\mathbf{y}}^{(l)})^{\mathrm{T}} H \tilde{\mathbf{y}}^{(l)} \in \mathbb{R}$: it is evaluated by $\frac{1}{\|\mathbf{y}^{(l)}\|^2}\Big((\mathbf{c}^{(l)})^{\mathrm{T}}\mathbf{b}^{(l)} - 2\sum_{i \in \mathcal{I}^{(l)}} c_i^{(l)} b_i^{(l)} + \sum_{i,j \in \mathcal{I}^{(l)}} c_i^{(l)} H_{i,j} c_j^{(l)}\Big)$
2. $((\tilde{\mathbf{y}}^{(l)})^{\mathrm{T}} H \mathcal{E}_{\mathcal{I}^{(l)}})^{\mathrm{T}} = \mathcal{E}_{\mathcal{I}^{(l)}}^{\mathrm{T}} H \tilde{\mathbf{y}}^{(l)} \in \mathbb{R}^k$: its i-th entry is evaluated by $\frac{1}{\|\mathbf{y}^{(l)}\|}\Big(b_i^{(l)} - \sum_{j \in \mathcal{I}^{(l)}} H_{i,j} c_j^{(l)}\Big)$
3. $\mathcal{E}_{\mathcal{I}^{(l)}}^{\mathrm{T}} H \mathcal{E}_{\mathcal{I}^{(l)}} \in \mathbb{R}^{k \times k}$: it is the submatrix of H with index set $\mathcal{I}^{(l)}$. For $i, j \in \mathcal{I}^{(l)}$, the entry $H_{i,j}$ is evaluated on the fly by $H_{i,j} = \langle D_i | \hat{\mathrm{H}} | D_j \rangle$.

In the other case when $\|\mathbf{y}^{(l)}\| = 0$, only the matrix block $\mathcal{E}_{\mathcal{I}^{(l)}}^{\mathrm{T}} H \mathcal{E}_{\mathcal{I}^{(l)}}$ is nonzero.

When k is not too large, the smallest eigenpair of $(Q^{(l)})^{\mathrm{T}} H Q^{(l)}$ can be easily retrieved by default LAPACK eigensolvers[21] for selected eigenvalues. Once this is complete, $\mathbf{z}^{(l)}$ is computed according to eq 17. By eq 14, we have

$$\gamma^{(l)} = \begin{cases} \dfrac{z_1^{(l)}}{\|\mathbf{y}^{(l)}\|}, & \text{if } \|\mathbf{y}^{(l)}\| \neq 0, \\ 1, & \text{otherwise} \end{cases} \tag{19}$$

and

$$\mathbf{a}^{(l)} = \mathbf{z}_{2:}^{(l)} - \gamma^{(l)}(\mathcal{E}_{\mathcal{I}^{(l)}})^{\mathrm{T}}\mathbf{c}^{(l)} \tag{20}$$

where $z_1^{(l)}$ denotes the first entry of $\mathbf{z}$, and $\mathbf{z}_{2:}^{(l)} \in \mathbb{R}^k$ the rest of the vector. Note that for the case of $\|\mathbf{y}^{(l)}\| = 0$, $z_1^{(l)}$ would always be zero, and $\gamma^{(l)}$ can be any value. For simplicity, we just set $\gamma^{(l)} = 1$.

2.2.3. Variable Update and Coefficient Compression. After obtaining the scaling factor $\gamma^{(l)}$ and the stepsize $\mathbf{a}^{(l)}$ from the procedure described in Section 2.2.2, vectors $\mathbf{c}$ and $\mathbf{b}$ are updated by

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

$$\mathbf{c}^{(l+1)} \leftarrow \gamma^{(l)}\mathbf{c}^{(l)} + \mathcal{E}_{\mathcal{I}^{(l)}}\mathbf{a}^{(l)} \tag{21}$$

and

$$\mathbf{b}^{(l+1)} \leftarrow \gamma^{(l)}\mathbf{b}^{(l)} + H\mathcal{E}_{\mathcal{I}^{(l)}}\mathbf{a}^{(l)} \tag{22}$$

It should be noted that in the actual implementation, $\mathbf{c}^{(l+1)}$ is not updated exactly as shown above. Instead, we store the scaling factor γ and modify only the relevant entries of $\mathbf{c}^{(l)}$, while most entries remain unchanged. For a detailed description of the implementation, please refer to Section 3.

Through the above update rules (eq 21) and (eq 22), the number of nonzero elements in $\mathbf{c}$ and $\mathbf{b}$ will continue to grow. Specifically, the growth rate for $\mathbf{b}$ is much faster than $\mathbf{c}$ in the beginning, because the number of entries involved in updating $\mathbf{b}$ in (eq 22) is much greater than updating $\mathbf{c}$ in (eq 21). This process can be seen as continuously adding determinants to the variational space, the selected subspace of the full CI space where we approximate the ground state. If we continue to iterate without compression, the variational space will eventually expand to the full CI space, which is obviously not feasible for large systems. To this end, we propose a compression strategy that targets $\mathbf{b}$. The update of $\mathbf{b}^{(l+1)}$ for a certain coordinate i will be discarded if both of the following conditions are satisfied: (i) if $b_i^{(l)} = 0$, indicating that the determinant $|D_i\rangle$ has not been selected before; (ii) $\Delta b_i^{(l+1)} \leq \tau$, with $\mathbf{\Delta b}^{(l+1)} := H\mathcal{E}_{\mathcal{I}^{(l+1)}}\mathbf{a}^{(l+1)}$, meaning that the update is relatively small. For any determinant $|D_i\rangle$ deemed unimportant, as long as both b_i and c_i remain zero, it will not be selected during the coordinate selection step. Thus, the size of variational space is controlled, and the wave function vector will converge into a subspace of the full CI space, consisting only of limited number of important determinants.

Notably, despite compressing the vector $\mathbf{b}$, b_i remains accurate for every determinant $|D_i\rangle$ with nonzero coefficients in $\mathbf{c}$, meaning it was selected at least once during the coordinate selection step. This is achieved by recalculating $b_i^{(l+1)}$ for each selected determinant $|D_i\rangle$ selected at the l-th iteration, using the following formula:

$$b_i^{(l+1)} = H_{i,:}\mathbf{c}^{(l+1)} = \sum_{j\in\mathcal{I}_H(i)} H_{i,j}c_j^{(l+1)} \tag{23}$$

Once a determinant $|D_i\rangle$ is selected and recalculated, updates to b_i will not be discarded again according to the compression rule. Therefore, $\mathbf{b}$ is only compressed for determinants that have not yet been added to the variational space. This ensures that the Rayleigh quotient $(\mathbf{c}^{\mathrm{T}}H\mathbf{c})/(\mathbf{c}^{\mathrm{T}}\mathbf{c})$, is accurately maintained in each iteration by

$$\frac{(\mathbf{c}^{(l)})^{\mathrm{T}}H\mathbf{c}^{(l)}}{(\mathbf{c}^{(l)})^{\mathrm{T}}\mathbf{c}^{(l)}} = \frac{(\mathbf{c}^{(l)})^{\mathrm{T}}\mathbf{b}^{(l)}}{(\mathbf{c}^{(l)})^{\mathrm{T}}\mathbf{c}^{(l)}}$$

since the compressed terms b_i always have associated $c_i = 0$ and therefore do not contribute to the sum.

Both the numerator and denominator are stored in memory, and updated in each iteration via the following straightforward update rules:

$$(\mathbf{c}^{(l+1)})^{\mathrm{T}}\mathbf{c}^{(l+1)} \leftarrow (\gamma^{(l)})^2(\mathbf{c}^{(l)})^{\mathrm{T}}\mathbf{c}^{(l)} + 2\gamma^{(l)}(\mathbf{a}^{(l)})^{\mathrm{T}}(\mathcal{E}_{\mathcal{I}^{(l)}}\mathbf{c}^{(l)}) + (\mathbf{a}^{(l)})^{\mathrm{T}}\mathbf{a}^{(l)} \tag{24}$$

and

$$(\mathbf{c}^{(l+1)})^{\mathrm{T}}\mathbf{b}^{(l+1)} \leftarrow (\gamma^{(l)})^2(\mathbf{c}^{(l)})^{\mathrm{T}}\mathbf{b}^{(l)} + 2\gamma^{(l)}(\mathbf{a}^{(l)})^{\mathrm{T}}(\mathcal{E}_{\mathcal{I}^{(l)}}\mathbf{b}^{(l)}) + (\mathbf{a}^{(l)})^{\mathrm{T}}H\mathbf{a}^{(l)} \tag{25}$$

2.2.4. Proposed Multicoordinate Descent FCI Method. The following outlines the complete procedure for the multicoordinate descent FCI method.

1. Initialize $\mathcal{I}^{(0)}$, the set of determinants preselected before computation begins. In our case, we choose $\mathcal{I}^{(0)} = \{|D_{\mathrm{HF}}\rangle\}$, which introduces only one nonzero entry in $\mathbf{c}^{(1)}$, simplifying the computation of $\mathbf{b}^{(1)}$. Set $l = 1$, initialize $\mathbf{c}^{(1)}$, and compute $\mathbf{b}^{(1)}=H\mathbf{c}^{(1)}$. Other initial vectors may be chosen, provided that $(\mathbf{c}^{(1)})^{\mathrm{T}}H\mathbf{c}^{(1)} < 0$ holds.
2. Select the index set $\mathcal{I}^{(l)} = \{i_1^{(l)}, ..., i_k^{(l)}\}$ according to (eq 10).
3. Evaluate the matrix $(Q^{(l)})^{\mathrm{T}}HQ^{(l)}$ following (eq 18). Compute its lowest eigenvalue and corresponding eigenvector $\mathbf{z}^{(l)}$, and then obtain the scaling factor $\gamma^{(l)}$ using (eq 19) and the stepsize vector $\mathbf{a}^{(l)}$ via (eq 20).
4. Update $\mathbf{c}^{(l+1)}$, $\mathbf{b}^{(l+1)}$, $(\mathbf{c}^{(l+1)})^{\mathrm{T}}\mathbf{c}^{(l+1)}$ and $(\mathbf{c}^{(l+1)})^{\mathrm{T}}\mathbf{b}^{(l+1)}$ according to (eq 21), (eq 22), (eq 24) and (eq 25). Use the compression technique based on a predefined threshold τ. Recalculate $b_i^{(l+1)}$ for $i \in \mathcal{I}^{(l)}$ as in (eq 23).
5. Repeat steps 2−4 with $l \leftarrow l + 1$ until either a stopping criterion or the maximum iteration number is reached. In our implementation, we adopt the stopping criterion proposed in OptOrbFCI.[22] The moving average S of the stepsize norm is tracked using a decay factor of $\alpha = 0.99$,

$$S^{(l+1)} \leftarrow \alpha S^{(l)} + (1 - \alpha)\left\|\mathbf{a}^{(l+1)}\right\|$$

The algorithm halts when $S^{(l+1)}$ falls below a specified tolerance level. Other stopping criteria, such as monitoring the change of the Rayleigh quotient or the ratio of the number of nonzero coefficients in $\mathbf{b}$ and $\mathbf{c}$,[17] may also be applied.

3. COMPUTATIONAL IMPLEMENTATION AND COMPLEXITY ANALYSIS

We will now provide implementation details of the algorithm, focusing on computationally expensive components and numerical stability issues, while conducting complexity analysis in the same time. In the following analysis, we denote $N_H = \max_i \mathcal{I}_H(i)$ as the maximum number of nonzero entries in columns of the Hamiltonian matrix, which scales as $O(n_{\mathrm{elec}}^2 n_{\mathrm{orb}}^2)$. We claim that the per-iteration complexity of our algorithm is $O(kN_H + k^3)$, and $O(N_H + k^3)$ for each thread with shared memory parallelism.

The sparse vectors $\mathbf{c}$ and $\mathbf{b}$ are stored in a hash table that allows parallel read/write operations[23] as in previous work.[17] For each entry of the hash table, its key is the Slater determinant $|D_i\rangle$ encoded by an n_{orb}-bit binary string, and its value is a pair of double floating-point numbers c_i and b_i. Three additional scalars are also stored in memory and updated in each iteration: the inner products $\mathbf{c}^{\mathrm{T}}\mathbf{c}$, $\mathbf{c}^{\mathrm{T}}\mathbf{b}$ and the scaling factor γ. All the three quantities are stored in quadruple-precision floating point format to mitigate accumulated numerical errors, ensuring accuracy unless the number of iterations exceeds 10^{16}.[17]

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

Note that the Hamiltonian matrix is never stored in memory: indeed, we evaluate each Hamiltonian entry on-the-fly. Different from other selected CI methods, this significantly reduces the storage cost of our algorithm, and allows us to utilize all available storage capacity to store the ground-state coefficient vector $\mathbf{c}$ and its associated $\mathbf{b} = H\mathbf{c}$ related to the residual.

Our implementation utilizes shared memory parallelism based on OpenMP. Specifically, we enable k threads where k is the number of coordinates we choose to update in each iteration.

In the coordinate pick stage (described in Section 2.2.1), each thread handling determinant $|D_i\rangle$ examines determinants in $\mathcal{I}_H(i)$ to select k determinants with the largest value of $[\nabla f(\mathbf{c})]_i$. This step has a complexity of $O(N_H)$ for each thread, and $O(kN_H)$ in total. After this, the main thread selects the k largest determinants from k^2 gathered from all the threads.

In the subproblem solve stage (described in Section 2.2.2), we begin by constructing the $(k+1)$-dimensional matrix $(Q^{(l)})^{\mathrm{T}}HQ^{(l)}$ in (eq 18). It is evident that computing $(\tilde{\mathbf{y}}^{(l)})^{\mathrm{T}}H\tilde{\mathbf{y}}^{(l)}$ and $\mathcal{E}^{\mathrm{T}}_{\mathcal{I}^{(l)}}H\tilde{\mathbf{y}}^{(l)}$ involves only $O(k^2)$ cost, whereas for $\mathcal{E}^{\mathrm{T}}_{\mathcal{I}^{(l)}}H\mathcal{E}_{\mathcal{I}^{(l)}}$, we need to evaluate k^2 Hamiltonian entries, each with complexity $O(n_{\mathrm{orb}})$. For the subsequent eigenvalue problem, we want to retrieve the smallest eigenvalue and its corresponding eigenvector of a $(k+1)$-dimensional symmetric matrix. A robust eigensolver like MRRR[21] yields such eigenpair at a cost of $O(k^3 + Tk^2)$ where T is the number of iteration steps, which is usually comparable to k.

Unfortunately, MRRR only guarantees that the extreme eigenvalue is converged to double precision, without guaranteeing the accuracy of each entry in the eigenvector. This could be problematic because the magnitudes of coordinates in the eigenvector can vary widely, and the small values of the stepsize vector might only have an accuracy of 10^{-8}. To address this issue, we employ one step of inverse iteration conducted in quadruple precision to improve the accuracy of eigenvector, which again has a complexity of $O(k^3)$.

Finally, during the variable update stage (described in Section 2.2.3), each thread is assigned to calculate a local update of $\mathbf{b}$ in the form $\Delta\mathbf{b}^i = H_{:,i}a_i$. Compression occurs at the thread level, skipping updates of coordinate j if $b_j = 0$ and $\Delta\mathbf{b}^i_j \le \tau$. In this process, each thread constructs $H_{:,i}$, the i-th column of H, which is subsequently used to recalculate $b_i^{(l+1)}$ as described in (eq 23). The complexity of the variable update step remains $O(N_H)$ per thread, resulting in a total complexity of $O(kN_H)$. The prefactor is dominated by the computation cost of a single Hamiltonian entry, which scales with $O(n_{\mathrm{orb}})$.

In the proposed algorithm, two synchronization points occur in each iteration: one after coordinate selection and another after the variable update. Specifically, update (eq 23) is performed sequentially following the parallel execution of (eq 21) and (eq 22), ensuring all updates are completed and maintaining data consistency across threads.

Table 1 gives a summary of computational cost for each step per thread and in total.

Table 1. Computational Cost for Each Step Per Thread and In Total[a]

Step	Complexity per thread	Complexity in total
Coordinate pick	$O(N_H + k^2)$	$O(kN_H + k^2)$
Subproblem solve	$O(k^3)$	$O(k^3)$
Variable update	$O(N_H)$	$O(kN_H)$

[a] N_H is defined as the maximum number of nonzero entries in any column of the Hamiltonian matrix, i.e., $N_H = \max_i \mathcal{I}_H(i)$.

4. NUMERICAL EXPERIMENTS

In this section, we conduct a series of numerical experiments to demonstrate the efficiency of the proposed algorithm—multicoordinate descent FCI method (mCDFCI). First, we compute the ground-state energy of N_2 using three different basis sets: cc-pVDZ, cc-pVTZ, and cc-pVQZ. As the number of orbitals increases, the calculated energies become more accurate, but the computational cost also rises. We test the computations using different numbers of cores and demonstrate the strong parallel scaling for big systems. Subsequently, we compare the convergence curve and computation time with the original CDFCI method, as well as other methods, namely SHCI, DMRG and iS-FCIQMC (initiator semistochastic FCIQMC). After that, we turn our eyes on a bigger system, the chromium dimer Cr_2 under the Ahlrichs SV basis at $r = 1.5$ Å, with 48 electrons and 84 spin−orbitals. This is a well-known multireference system because of the spin coupling of the 12 valence electrons. Finally, we benchmark the binding curve of nitrogen dimer under the cc-pVQZ basis, which consists of 220 spin−orbitals.

In all experiments, the orbitals and integrals are calculated via restricted Hartree−Fock (RHF) in PSI4 package[24] version 1.8. All energies are reported in the unit of Hartree (Ha).

The program is compiled using the GNU g++ compiler version 11.4.0 with the -O3 optimization flag and its native OpenMP support. It is linked against LAPACK version 3.11.0 and OpenBLAS version 0.3.26 for numerical linear algebra computations.

4.1. Scalability and Performance Testing. The N_2 molecule is considered a correlated system due to the significant role of electron correlation effects in accurately describing its electronic structure. We compute its ground-state energy using three different basis sets: cc-pVDZ (14e, 28o), cc-pVTZ (14e, 60o), and cc-pVQZ (14e, 110o). A detailed description of the system and the configurations used when running the CDFCI program is provided in Table 2. The experiments in this section were conducted on a system equipped with two AMD EPYC 9754 128-core processors and 1.5 TB of memory.

In the multithreaded implementation of CDFCI (mCDFCI), we define *effective iterations* as the product of the number of iterations and the number of determinants updated, ensuring that mCDFCI performs an equivalent

Table 2. Description of the Basis Sets, Configurations, Thresholds, and Average Nonzero per Column Used for N_2 in Section 4.1[a]

Basis Set	Electrons	Spin Orbitals	Threshold τ	N_H
cc-pVDZ	14	56	5×10^{-7}	$<4 \times 10^3$
cc-pVTZ	14	120	5×10^{-6}	$<3 \times 10^4$
cc-pVQZ	14	220	5×10^{-5}	$<8 \times 10^4$

[a] N_H is defined as the maximum number of nonzero entries in any column of the Hamiltonian matrix, i.e., $N_H = \max_i \mathcal{I}_H(i)$.

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

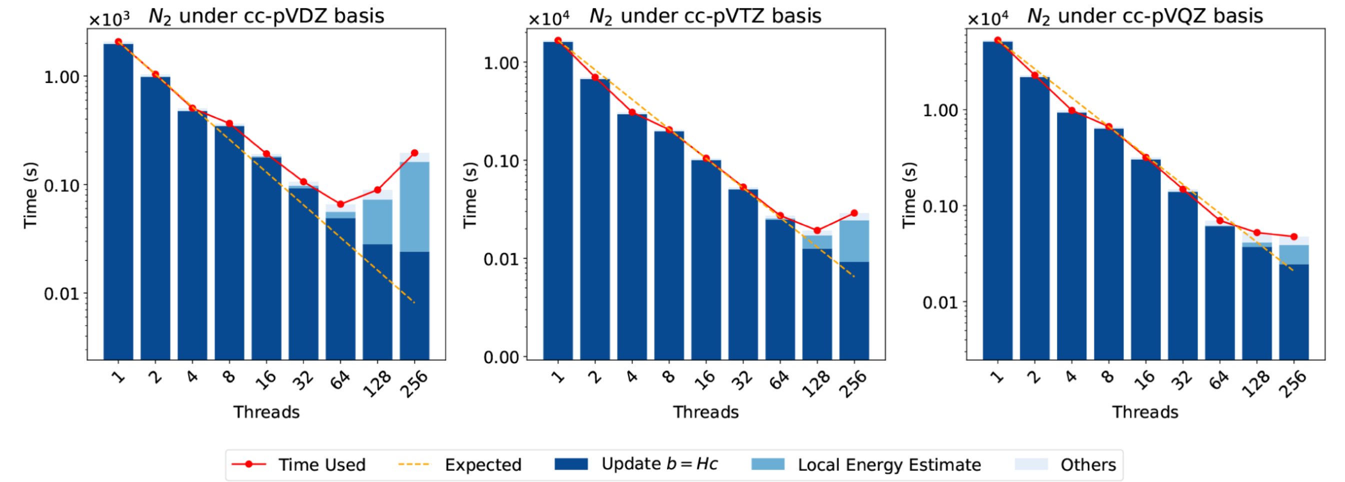


Figure 2. Comparison of runtime for the N_2 molecule with different basis sets (cc-pVDZ, cc-pVTZ, and cc-pVQZ) using various numbers of threads. Each subplot represents a different basis set. The bar charts show the time in seconds spent on specific computational steps, including updating $\mathbf{b} = H\mathbf{c}$, local energy estimation, and other operations. The red dots on the line plot show the total time usage, while the yellow dashed line represents the ideal scaling behavior (expected performance improvement with increased threads).

Table 3. Convergence of Ground-State Energy for C_2[a]

					mCDFCI	
Algorithm	Parameter	Absolute error	Mem (GB)	Time (s)	Time (s)	Ratio
mCDFCI	$\tau = 3.0 \times 10^{-8}$	1.0×10^{-2}	1.5	3.8	-	
		1.0×10^{-3}	6.0	17.2	-	
		1.0×10^{-4}	24.0	123.4	-	
		1.0×10^{-5}	48.0	603.7	-	
		1.0×10^{-6}	48.0	2332.3	-	
sCDFCI	$\varepsilon = 3.0 \times 10^{-8}$	1.0×10^{-2}	3.0	14.7	3.8	3.87 ×
		1.0×10^{-3}	6.0	55.0	17.2	3.20 ×
		1.0×10^{-4}	24.0	374.3	123.4	3.03 ×
		1.0×10^{-5}	48.0	1779.7	603.7	2.95 ×
		1.0×10^{-6}	48.0	6906.0	2332.3	2.96 ×
SHCI	$\varepsilon_1 = 1.0 \times 10^{-4}$	1.5×10^{-3}	16.1	3.3	14.0	0.23 ×
	$\varepsilon_1 = 5.0 \times 10^{-5}$	7.6×10^{-4}	17.5	7.3	23.7	0.31 ×
	$\varepsilon_1 = 1.0 \times 10^{-5}$	1.1×10^{-4}	32.6	53.9	113.2	0.48 ×
	$\varepsilon_1 = 5.0 \times 10^{-6}$	4.6×10^{-5}	54.2	126.0	219.4	0.57 ×
	$\varepsilon_1 = 1.0 \times 10^{-6}$	4.7×10^{-6}	238.1	845.0	954.1	0.89 ×
DMRG	max M = 500	6.9×10^{-4}	0.2	106.9	23.7	4.52 ×
	max M = 1000	1.5×10^{-4}	0.8	345.2	92.7	3.72 ×
	max M = 2000	1.9×10^{-5}	3.1	1130.1	394.9	2.86 ×
	max M = 4000	2.1×10^{-6}	11.5	3848.3	1549.2	2.48 ×
iS-FCIQMC	$m = 10000$	$7.4 \pm 1.6 \times 10^{-4}$	0.076	19.3	23.7	0.81 ×
	$m = 50000$	$6.4 \pm 0.6 \times 10^{-4}$	0.077	59.8	26.9	2.22 ×
	$m = 100000$	$1.8 \pm 0.4 \times 10^{-4}$	0.077	118.0	78.6	1.50 ×
	$m = 500000$	$2.3 \pm 0.2 \times 10^{-4}$	0.080	459.9	64.5	7.13 ×
	$m = 1000000$	$5.4 \pm 1.2 \times 10^{-5}$	0.083	872.4	196.5	4.44 ×

[a]The reference ground-state energy is −75.7319615 Ha.

workload regardless of the number of threads. Notably, in our experiments, while different values of k produce distinct energy descent trajectories, the energy values after the same number of effective iterations remain consistent, irrespective of the number of threads used. Specifically, for the cc-pVDZ basis, the maximum difference between energy values was 4.84×10^{-7} Ha, while for cc-pVTZ and cc-pVQZ, the differences were 2.66×10^{-6} Ha and 9.09×10^{-6} Ha, respectively. All of these differences are smaller than 10^{-5} Ha, confirming the robustness and stability of the method.

Figure 2 shows the runtime of mCDFCI over 1024K effective iterations, evaluated across a range of thread counts from 1 to 256. The detailed data supporting these results are provided in Appendix B, where the exact runtime of each component is presented, along with an in-depth analysis of the parallel capabilities. In every case, the dominant computational cost arises from the update of the vector $\mathbf{b} = H\mathbf{c}$. However, the time for the local energy estimation, which is not parallelized, increases as the thread count grows. This behavior can be attributed to the increasing impact of the k^3 term in the eigenvalue computation as k becomes larger. This increase is

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

Table 4. Convergence of Ground-State Energy for N_2[a]

Algorithm	Parameter	Absolute error	Mem (GB)	Time (s)	mCDFCI Time (s)	Ratio
mCDFCI	$\tau = 5.0 \times 10^{-7}$	1.0×10^{-2}	3.0	4.2	-	
		1.0×10^{-3}	12.0	31.8	-	
		1.0×10^{-4}	24.0	311.8	-	
		1.0×10^{-5}	24.0	1830.3	-	
		1.0×10^{-6}	24.0	5629.8	-	
sCDFCI	$\varepsilon = 5.0 \times 10^{-7}$	1.0×10^{-2}	6.0	19.2	4.2	4.57 ×
		1.0×10^{-3}	12.0	87.5	31.8	2.75 ×
		1.0×10^{-4}	24.0	829.7	311.8	2.66 ×
		1.0×10^{-5}	24.0	4687.7	1830.3	2.56 ×
		1.0×10^{-6}	24.0	14048.5	5629.8	2.50 ×
SHCI	$\varepsilon_1 = 1.0 \times 10^{-4}$	1.6×10^{-3}	16.8	4.0	19.7	0.20 ×
	$\varepsilon_1 = 5.0 \times 10^{-5}$	8.8×10^{-4}	18.9	10.9	40.0	0.27 ×
	$\varepsilon_1 = 1.0 \times 10^{-5}$	1.9×10^{-4}	37.9	76.2	175.2	0.44×
	$\varepsilon_1 = 5.0 \times 10^{-6}$	8.6×10^{-5}	68.2	183.4	353.3	0.52 ×
	$\varepsilon_1 = 1.0 \times 10^{-6}$	1.0×10^{-5}	402.7	1753.8	1796.5	0.98 ×
DMRG	max M = 500	1.2×10^{-3}	0.2	126.5	27.9	4.54 ×
	max M = 1000	3.1×10^{-4}	0.8	427.3	114.2	3.74 ×
	max M = 2000	6.2×10^{-5}	3.1	1405.8	470.7	2.99 ×
	max M = 4000	8.9×10^{-6}	11.8	4940.7	1976.8	2.50 ×
iS-FCIQMC	m = 10000	$4.4 \pm 1.4 \times 10^{-4}$	0.162	31.3	79.6	0.39 ×
	m = 50000	$2.4 \pm 0.5 \times 10^{-4}$	0.162	108.9	141.0	0.77 ×
	m = 100000	$1.2 \pm 0.3 \times 10^{-4}$	0.163	233.0	270.1	0.86 ×
	m = 500000	$7.5 \pm 1.4 \times 10^{-5}$	0.166	858.7	398.8	2.15 ×
	m = 1000000	$3.5 \pm 1.2 \times 10^{-5}$	0.169	1596.3	735.0	2.17 ×

[a]The reference ground-state energy is −109.2821721 Ha.

particularly noticeable for smaller systems, such as cc-pVDZ, where the computational time grows significantly beyond 64 threads.

For larger systems, strong parallel scalability is observed up to 128 threads, with instances of superlinear speedup. This performance gain is attributed to the hash table we use, which is specifically designed and optimized for multithreaded workloads. When focusing solely on the time taken to update the hash table and disregarding the local energy estimation, it is clear that the update of the **b** vector is highly parallelizable.

4.2. Numerical Results with C_2 and N_2. In this section, we compare the performance of our method with other FCI methods: SHCI, DMRG and iS-FCIQMC, all with parallel support and using the same amount of computing resources. We test all the algorithms on two systems: C_2 and N_2 under the cc-pVDZ basis sets. The original CDFCI program[25] which supports OpenMP acceleration by parallelizing the update $\mathbf{b} = H\mathbf{c}$, is also tested here and renamed as sCDFCI (single-coordinate descent FCI) for comparison with our new method mCDFCI (multicoordinate descent FCI). SHCI, DMRG and iS-FCIQMC calculations were performed using the Arrow code,[26] the Block2 code[27] and the NECI code,[28] respectively. All programs were compiled using the GNU g++ compiler (version 11.4.0) with the -O3 optimization flag. Native support for OpenMP was utilized, and MPI support was provided by MPICH (version 4.2.0). All programs were restricted to use 64 cores. CDFCI, SHCI, and DMRG were executed with 64 threads each, while iS-FCIQMC, which supports only MPI parallelization, was executed with 64 MPI processes. Apart from the parallelization configurations enabled in our study, we used the same experimental settings as described in Wang et al.[17].

Specifically, SHCI was run with different thresholds ε_1, DMRG with varying maximum bond dimensions M, and iS-FCIQMC with different numbers of walkers m. For iS-FCIQMC, the preconditioned heat-bath (PCHB) excitation generator and the semistochastic method (with 1000 determinants in the deterministic space) were employed. The initiator threshold was set to 3.0 and the number of iterations was set to 30,000 to ensure convergence by observation of clear plateaus in the diagnostic plots. The Hartree−Fock state was used as the initial wave function for all algorithms except DMRG, which has its own warm-up algorithm. In all cases, the energy is reported without any perturbation or extrapolation postcalculations. Variational energy is reported for CDFCI, SHCI and DMRG, while average projected energy is reported for iS-FCIQMC, along with stochastic error estimates obtained through additional blocking analysis. The absolute error is defined as the absolute difference between the reported energy and the reference ground-state energy.

Tables 3 and 4 demonstrate the convergence behavior of the C_2 and N_2 molecule respectively, executed by different methods. Each method was evaluated with specific parameter values, and the results are documented in terms of absolute error, memory consumption in gigabytes (GB), execution time in seconds, and a comparison with the mCDFCI algorithm regarding execution time and relative ratio. From a broader perspective, there are significant differences in time and memory usage among the various algorithms, which we will examine in detail.

The proposed algorithm, mCDFCI, achieves chemical accuracy for the C_2 molecule in less than 1 min while using only 6 GB of memory. Subsequently, the energy continues to decrease at a linear rate. The expansion of the primary

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

variational space slows down once the absolute error is less than 10^{-5}. The wave function converges to a subspace of the full CI space according to the current truncation threshold $\tau = 3 \times 10^{-8}$. For the N_2 molecule, mCDFCI exhibits similar behavior for threshold $\tau = 5 \times 10^{-7}$.

Compared to sCDFCI, which updates only one determinant per iteration, mCDFCI descends much faster and utilizes a smaller subspace at the start of the iterations. This demonstrates the efficiency of mCDFCI in capturing important determinants. Toward the end of the iterations, the effective iterations of mCDFCI and sCDFCI are comparable. However, mCDFCI still shows approximately three times the acceleration, highlighting the enhanced parallelization power utilized by our approach.

SHCI also adopts the concept of selected CI but employs a different data structure compared to ours. After selecting determinants based on an importance measure from the Hamiltonian matrix elements connected to the reference determinant, SHCI extracts a submatrix and directly diagonalizes it to find the smallest eigenpair. The efficiency of the SHCI algorithm can be attributed to the powerful Davidson algorithm, the one-time evaluation of the Hamiltonian matrix, and the contiguous memory read/write operations. Additionally, multithreading is particularly effective for computing matrix-vector multiplications, making the method highly parallel and further boosting the performance. These factors enable SHCI to exhibit speedups over mCDFCI, especially when the required accuracy is low. However, the storage of the Hamiltonian submatrix leads to high storage costs, particularly when high accuracy is required. On the other hand, when $\varepsilon_1 = 1 \times 10^5$, SHCI uses 3,324,630 determinants to reach an accuracy of 1.1×10^{-4} Ha, whereas mCDFCI uses only 2,022,447 determinants to achieve the same accuracy. In this case, mCDFCI uses approximately 40% fewer determinants than SHCI to attain the same level of precision.

DMRG, which models the wave function using a matrix product state form, has a lower memory footprint than mCDFCI due to its storage cost of $O(n_{\mathrm{orb}}M^2)$, where n_{orb} is the number of orbitals and M the maximum bond dimension. However, mCDFCI achieves faster convergence compared to DMRG, particularly when targeting moderate accuracy. The difference in convergence speed decreases as higher accuracy is required, but mCDFCI remains more efficient overall.

Finally, iS-FCIQMC, a stochastic method that approximates the wave function using random walkers, is highly memory-efficient, typically requiring a few megabytes. While its error decreases gradually as the number of walkers increases, iS-FCIQMC can achieve faster convergence for lower accuracy requirements. On the other hand, mCDFCI, with its deterministic framework, offers a reliable and efficient approach for achieving higher precision.

4.3. All-Electron Chromium Dimer Calculation. We now turn our attention to the larger system of the chromium dimer, which consists of 48 electrons and 84 spin−orbitals. Transition metals play a pivotal role in catalysis, biochemistry, and the energy sciences, but the complex electronic structure of d-shells presents significant challenges for modeling and understanding such processes.[29]

In the case of the chromium dimer, the high electron correlation arises from two main aspects: static correlation, which is due to the spin coupling of the 12 valence electrons, and dynamic correlation, which involves excitations of nonvalence orbitals, such as the 3p electrons. To address this challenge, we computed the all-electron chromium dimer ground state at a bond length of 1.5 Å using the Ahlrichs SV basis set.

Figure 3 illustrates the energy decrease and time usage for running 2×10^7 iterations using 128 threads. In this setup, 128

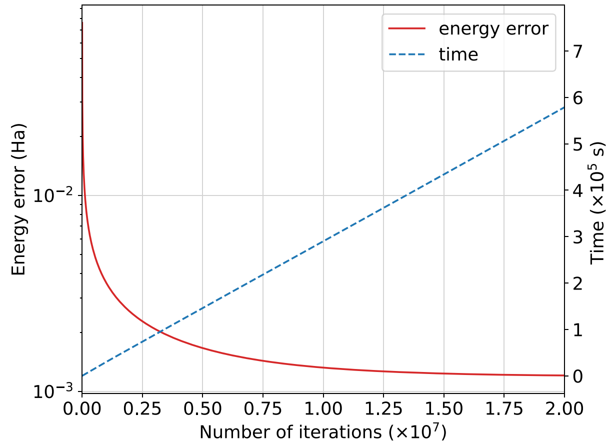


Figure 3. Convergence of ground-state energy for the Cr_2 molecule under Ahlrichs SV basis. The threshold is set to 3.7×10^{-5} and the reference ground-state energy is −2086.444784 Ha, obtained via extrapolated DMRG.[30]

determinants are selected in each iteration, for either addition to the variational space or updating their associated values. The final number of determinants in the variational wave function is approximately 2×10^9.

The red curve in the plot represents the absolute energy error (in Hartree) as a function of the number of iterations. The energy error is computed as the absolute difference between the current variational energy (Rayleigh quotient) and the reference ground-state energy. This error decreases rapidly to the level of 10^{-3} within several hours, before converging toward the ground-state energy at a steady linear rate.

The blue dashed curve in the plot shows the total time elapsed as a function of the number of iterations, measured in seconds ($\times 10^5$). A total of 2×10^7 iterations takes around 6×10^5 seconds, which is roughly 1 week. The linear trend indicates that the time per iteration remains relatively constant. This consistency is due to the number of nonzeros in each column of the Hamiltonian matrix being similar (around 3×10^4), leading to a nearly constant computational cost per iteration (See Table 1 for complexity analysis). This predictable linear increase in time per iteration allows users to estimate the remaining computation time easily, demonstrating the practicality of the CDFCI algorithm.

Compared to the previous CDFCI algorithm with single-coordinate updates, which achieved an energy of −2086.443565 Ha after one month of computation,[17] we reached the same level of accuracy in just 5.8 days, despite using different computing environments. Additionally, we obtained a more accurate energy of −2086.443581 Ha in approximately 8.8 days.

4.4. Binding Curve of N_2 Under Cc-pVQZ Basis. Finally, we benchmark the all-electron nitrogen binding curve under cc-pVQZ basis. The problem is challenging due to the complexity of accurately capturing the strong electron correlation in the triple bond of N_2. The large cc-pVQZ

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

basis set increases computational cost, and all-electron calculations add further complexity by requiring the explicit treatment of core electrons. We adopt $\tau = 5 \times 10^{-6}$ for mCDFCI truncation, use 128 threads, and run 1,000,000 iterations. The bond length is varied from 1.5 to 4.5 a_0. Each configuration on the binding curve takes roughly 18 h to achieve chemical accuracy.

Figure 4 displays a smooth, standard-shaped binding curve (shown in red) for N_2 calculated using the mCDFCI method

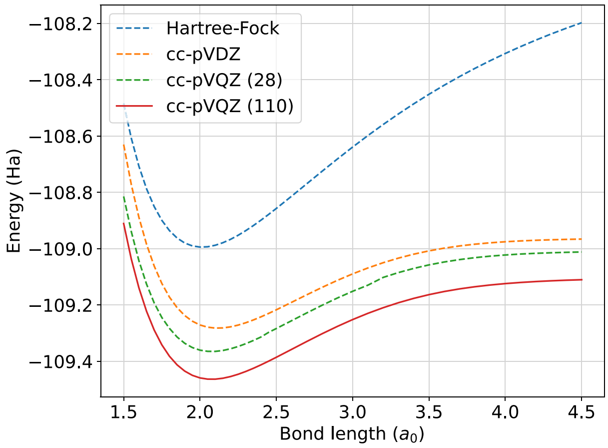


Figure 4. Ground-state energy versus bond length for N_2 using different basis sets. cc-pVQZ (110) refers to the full, uncompressed cc-pVQZ basis set, while cc-pVQZ (28) refers to the compressed set of orbitals generated by OptOrbFCI. The mCDFCI truncation threshold $\tau = 5 \times 10^{-6}$ and stopping criteria of 1×10^{-5} are applied.

with a full cc-pVQZ basis set (110 orbitals). The curve reaches its minimum at the equilibrium bond length of $r = 2.118\ a_0$, which corresponds to the optimal bond length where the energy is minimized. This red curve is compared against other computational results, including the Hartree−Fock approximation (blue dashed line) and CDFCI calculations using a reduced orbital set generated from cc-pVQZ, which is compressed to 28 selected orbitals (green dashed curve).

The Hartree−Fock method overestimates the total energy, particularly at longer bond lengths, due to its inability to capture dynamic electron correlation and its inherent inclusion of ionic character.[31] The green dashed curve represents results from the OptOrbFCI method,[22] which uses a rotation and compression of the full cc-pVQZ basis set into 28 selected orbitals. This method was employed to reduce the computational cost while retaining the essential characteristics of the electronic structure of the system. Despite the reduction in the number of orbitals, OptOrbFCI produces an energy profile closer to the full cc-pVQZ calculation than the cc-pVDZ approximation, showing that the selected orbitals capture most of the correlation effects. However, the slight increase in energy compared to the full cc-pVQZ curve reflects the inherent trade-off between computational efficiency and accuracy when compressing the basis set.

Appendix C lists all the variational energies corresponding to each bond length and configuration shown in the figure, providing a detailed comparison of the methods and basis sets. This comparison underscores the importance of increasing the basis set size to achieve basis set convergence in quantum chemical calculations.

5. CONCLUSION

This paper presents a novel algorithm, multicoordinate descent FCI method, for solving the full configuration interaction (FCI) problem in electronic structure calculations. The algorithm reformulates the FCI eigenvalue problem as an unconstrained optimization problem and employs a multicoordinate descent approach to update the optimization variable. Additionally, the algorithm introduces a compression strategy to control the size of the variational space.

The method demonstrates efficiency and accuracy through various benchmarks, including the computation of an accurate benchmark energy for the chromium dimer and the binding curve of nitrogen dimer. The results show up to 79.3% parallel efficiency on 128 cores, as well as significant speedup compared to previous methods. Overall, the multicoordinate descent FCI method presents a promising approach for solving the FCI problem in electronic structure calculations, with improved efficiency, accuracy, and parallelization capability, making it suitable for larger systems and more accurate approximations.

Since the current algorithm focuses on improving the variational stage of selected CI, our immediate future work will involve adding a perturbation stage to the current algorithm. Another direction will be to combine the current results with orbital optimization, as it is well-known that optimized orbitals lead to faster convergence.[22,32,33] Several approaches for computing excited states within the FCI framework have been developed.[34−38] These methods, often extensions of ground-state algorithms, have demonstrated success in accurately treating excited states. Recently, Wang et al. introduced xCDFCI,[39] an efficient extension of CDFCI designed specifically for low-lying excited states in molecules. We aim to adapt our techniques for excited-state computations as well.

Regarding parallel capacity, extending our current shared-memory implementation to a distributed-memory one with MPI enabled is an interesting prospect, as it would further increase computational capacity and leverage memory load. A well-implemented distributed memory setup is under investigation.

■ APPENDIX A

Proof in Section 2.2.2

We prove $\lambda_{min}((Q^{(l+1)})^{\mathrm{T}}HQ^{(l+1)}) < 0$ as long as we choose an initial vector $\mathbf{c}^{(1)}$ such that $(\mathbf{c}^{(1)})^{\mathrm{T}}H\mathbf{c}^{(1)} < 0$, where $Q^{(l+1)}$ is defined in eq 13. The proof uses induction and contradiction. In simple terms, if we assume $(Q^{(l+1)})^{\mathrm{T}}HQ^{(l+1)}$ is positive semi-definite, then the minimization problem will attain its minimum when the optimization vector is zero. However, we can always find a vector that leads to a smaller value, thus contradicting our initial assumption.

We first show that if $\mathbf{c}^{(1)}$ is chosen such that $(\mathbf{c}^{(1)})^{\mathrm{T}}H\mathbf{c}^{(1)} < 0$, then, starting from the first iteration, we have $\lambda_{min}((Q^{(1)})^{\mathrm{T}}HQ^{(1)}) < 0$. Conversely, if all eigenvalues of $(Q^{(1)})^{\mathrm{T}}HQ^{(1)}$ are non-negative, solving the minimization problem (eq16) yields $\mathbf{z}^{(1)} = \mathbf{0}$ and $f(\mathbf{z}^{(1)}) = \|H\|_{\mathrm{F}}^2$. However, $\|H\|_{\mathrm{F}}^2$ is clearly not the minimal point for $f(\mathbf{c}^{(2)})$. To see this, we take

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

$$\gamma^{(1)} = \frac{\sqrt{-(\mathbf{c}^{(1)})^{\mathrm{T}} H \mathbf{c}^{(1)}}}{(\mathbf{c}^{(1)})^{\mathrm{T}} \mathbf{c}^{(1)}}$$

and $\mathbf{a}^{(1)} = \mathbf{0}$, we have

$$\begin{aligned} f(\mathbf{c}^{(2)}) &= \left\| H + \gamma^{(1)}\mathbf{c}^{(1)}(\gamma^{(1)}\mathbf{c}^{(1)})^{\mathrm{T}} \right\|_{\mathrm{F}}^2 \\ &= \left\| H \right\|_{\mathrm{F}}^2 + 2(\gamma^{(1)})^2(\mathbf{c}^{(1)})^{\mathrm{T}} H \mathbf{c}^{(1)} + (\gamma^{(1)})^4 ((\mathbf{c}^{(1)})^{\mathrm{T}}\mathbf{c}^{(1)})^2 \\ &= \left\| H \right\|_{\mathrm{F}}^2 - \left(\frac{(\mathbf{c}^{(1)})^{\mathrm{T}} H \mathbf{c}^{(1)}}{(\mathbf{c}^{(1)})^{\mathrm{T}}\mathbf{c}^{(1)}} \right)^2 < \left\| H \right\|_{\mathrm{F}}^2 \end{aligned}$$

Thus, $(Q^{(1)})^{\mathrm{T}} H Q^{(1)}$ must have at least one negative eigenvalue.

We then show that if, in the previous step, we have $\lambda_{min}((Q^{(l)})^{\mathrm{T}} H Q^{(l)}) < 0$, it follows that $\lambda_{min}((Q^{(l+1)})^{\mathrm{T}} H Q^{(l+1)}) < 0$. The reasoning is similar to above. Suppose this is not true, such that all eigenvalues of $(Q^{(l+1)})^{\mathrm{T}} H Q^{(l+1)}$ are non-negative. Solving the minimization problem (eq 16) would result in $\mathbf{z}^{(l+1)} = \mathbf{0}$ and $f(\mathbf{z}^{(l+1)}) = \left\| H \right\|_{\mathrm{F}}^2$. However, this is impossible. To see why, consider taking $\gamma^{(l+1)} = 1$ and $\mathbf{a}^{(l+1)} = \mathbf{0}$. In this case, the function $f(\mathbf{c}^{(l+1)})$ would have the same value as in the previous iteration:

$$\begin{aligned} f(\mathbf{c}^{(l+1)}) &= \left\| H + \mathbf{c}^{(l)}(\mathbf{c}^{(l)})^{\mathrm{T}} \right\|_{\mathrm{F}}^2 \\ &= \left\| H \right\|_{\mathrm{F}}^2 - (\lambda_{\min}((Q^{(l)})^{\mathrm{T}} H Q^{(l)}))^2 < \left\| H \right\|_{\mathrm{F}}^2 \end{aligned}$$

This demonstrates that the objective function value must decrease in each iteration. Additionally, in each iteration, we must have $\lambda_{min}((Q^{(l+1)})^{\mathrm{T}} H Q^{(l+1)}) < 0$.

Appendix B

Ground-State Energy Computation Time of N_2 Using Different Basis Sets and Different Number of Threads. Tables 5−7 present the mCDFCI computation times for the N_2 molecule across varying number of threads, using the cc-pVDZ, cc-pVTZ, and cc-pVQZ basis sets, respectively. These results are already illustrated in Figure 2. Here, we provide the corresponding parallel efficiency and a detailed time breakdown for each computational step.

Table 5. Time, Parallel Efficiency, and Component Breakdown for Ground-State Energy of N_2 Using cc-pVDZ (14e, 28o)

Threads	Time (s)	Parallel efficiency	Variable update (s)	Local energy update (s)	Others (s)
1	2076.40	1.000	1974.46	12.28	89.29
2	1038.48	1.000	980.75	7.67	49.66
4	505.32	1.027	474.03	4.78	26.33
8	365.66	0.710	345.40	3.84	16.26
16	192.09	0.676	178.04	3.85	10.11
32	106.10	0.612	92.77	4.94	8.34
64	65.91	0.492	49.03	6.99	9.82
128	89.31	0.182	28.07	44.65	16.54
256	195.71	0.041	24.09	138.11	33.45

Table 6. Time, Parallel Efficiency, and Component Breakdown for Ground-State Energy of N_2 Using cc-pVTZ (14e, 60o)

Threads	Time (s)	Parallel efficiency	Variable update (s)	Local energy update (s)	Others (s)
1	16626.01	1.000	16074.56	22.67	528.26
2	7003.45	1.187	6720.30	11.92	270.99
4	3084.88	1.347	2939.85	7.61	137.25
8	2045.62	1.016	1967.34	5.10	73.07
16	1045.78	0.994	1001.69	4.36	39.65
32	531.91	0.977	502.56	5.42	23.86
64	272.70	0.953	247.74	7.39	17.50
128	192.42	0.675	125.78	45.50	21.09
256	289.45	0.224	91.93	152.79	44.68

Table 7. Time, Parallel Efficiency, and Component Breakdown for Ground-State Energy of N_2 Using cc-pVQZ (14e, 110o)

Threads	Time (s)	Parallel efficiency	Variable update (s)	Local energy update (s)	Others (s)
1	53182.43	1.000	51319.96	36.06	1825.70
2	22897.89	1.161	21927.33	26.10	943.89
4	9834.53	1.352	9347.02	10.28	477.05
8	6669.42	0.997	6358.72	9.77	300.77
16	3190.37	1.042	3031.49	8.08	150.68
32	1485.38	1.119	1393.64	6.90	84.75
64	699.20	1.188	610.01	8.84	80.29
128	524.26	0.793	371.17	43.86	109.17
256	475.64	0.437	245.80	142.78	87.00

The time required for variable updates decreases steadily as the number of threads increases, demonstrating the strong scalability of the multi-threaded implementation. Conversely, the other two non-parallelized components initially exhibit a decrease in time, followed by an eventual increase. The initial decrease can be attributed to the reduced number of iterations required as the number of coordinates grows, given that when k is small, the time to solve a $(k+1)\times(k+1)$ eigenvalue problem remains relatively constant. However, as the number of coordinates continues to increase, the k^3 scaling in the eigenvalue computation becomes increasingly significant, resulting in a noticeable rise in the time required for local energy updates. To address this bottleneck, parallelizing these components represents an important area for future development of the CDFCI software package.

Appendix C

N_2 Binding Curve Data for Bond Length and Energy. Table 8 shows the binding curve data for four sets of calculations: Hartree−Fock (HF), CDFCI with the cc-pVDZ basis, CDFCI with the full cc-pVQZ (110 orbitals), and CDFCI with a reduced set of 28 compressed orbitals derived from the cc-pVQZ basis using the OptOrbFCI method.[22] The energies for HF were generated using the PSI4 package[24] version 1.8. The energies for the cc-pVDZ and cc-pVQZ (28) basis sets are cited from previous studies.[17,22] The table includes the bond lengths in atomic units (a_0) and the corresponding ground-state energies in Hartree (Ha).

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337

Table 8. Binding Curve Data for Bond Length and Energy, Showing the Results from Hartree−Fock (HF), cc-pVDZ, and CDFCI with Both the Full cc-pVQZ Basis (110 Orbitals) and the Compressed cc-pVQZ Basis (28 Orbitals)

Bond length (a_0)	Hartree−Fock	cc-pVDZ	cc-pVQZ (28)	cc-pVQZ (110)
1.50	−108.484166	−108.630048	−108.814403	−108.910666
1.55	−108.607446	−108.771997	−108.939182	−109.036535
1.60	−108.707310	−108.888846	−109.042905	−109.139272
1.65	−108.787292	−108.984314	−109.126015	−109.222403
1.70	−108.850397	−109.061575	−109.192655	−109.288927
1.75	−108.899180	−109.123348	−109.244370	−109.341398
1.80	−108.935822	−109.171964	−109.283011	−109.381998
1.85	−108.962179	−109.209426	−109.313701	−109.412578
1.90	−108.979837	−109.237458	−109.335975	−109.434729
1.95	−108.990152	−109.257541	−109.351156	−109.449809
2.00	−108.994283	−109.270953	−109.360360	−109.458975
2.05	−108.993221	−109.278790	−109.364582	−109.463219
2.10	−108.987816	−109.281994	−109.364756	−109.463348
2.15	−108.978797	−109.281374	−109.361496	−109.460105
2.20	−108.966786	−109.277621	−109.355553	−109.454133
2.25	−108.952320	−109.271328	−109.347340	−109.445965
2.30	−108.935857	−109.263001	−109.337391	−109.436048
2.35	−108.917789	−109.253072	−109.325979	−109.424753
2.40	−108.898453	−109.241908	−109.313533	−109.412410
2.45	−108.878135	−109.229823	−109.297016	−109.399302
2.50	−108.857081	−109.217083	−109.283533	−109.385670
2.60	−108.813564	−109.190508	−109.255799	−109.357597
2.70	−108.769211	−109.163600	−109.227940	−109.329427
2.80	−108.724949	−109.137358	−109.200871	−109.302021
2.90	−108.681424	−109.112473	−109.175147	−109.275997
3.00	−108.639072	−109.089405	−109.151280	−109.251749
3.10	−108.598176	−109.068450	−109.129515	−109.229566
3.20	−108.558912	−109.049779	−109.102007	−109.209619
3.30	−108.521376	−109.033462	−109.084817	−109.191894
3.40	−108.485608	−109.019484	−109.070067	−109.176446
3.50	−108.451609	−109.007747	−109.057557	−109.163193
3.60	−108.419352	−108.998083	−109.047199	−109.152036
3.70	−108.388793	−108.990269	−109.038765	−109.142780
3.80	−108.359873	−108.984050	−109.031901	−109.135188
3.90	−108.332527	−108.979162	−109.026548	−109.129007
4.00	−108.306685	−108.975357	−109.022348	−109.123980
4.10	−108.282276	−108.972410	−109.018986	−109.119960
4.20	−108.259228	−108.970132	−109.016448	−109.116684
4.30	−108.237469	−108.968366	−109.014495	−109.114082
4.40	−108.216928	−108.966991	−109.012920	−109.111935
4.50	−108.197538	−108.965910	−109.011722	−109.110256

AUTHOR INFORMATION

Corresponding Authors

Weiguo Gao − *School of Mathematical Sciences and School of Data Science, Fudan University, Shanghai 200433, China; Shanghai Key Laboratory for Contemporary Applied Mathematics, Shanghai 200433, China; Key Laboratory of Computational Physical Sciences (MOE), Shanghai 200433, China*; Email: wggao@fudan.edu.cn

Yingzhou Li − *School of Mathematical Sciences, Fudan University, Shanghai 200433, China; Shanghai Key Laboratory for Contemporary Applied Mathematics, Shanghai 200433, China; Key Laboratory of Computational Physical Sciences (MOE), Shanghai 200433, China*; orcid.org/0000-0003-1852-3750; Email: yingzhouli@fudan.edu.cn

Author

Yuejia Zhang − *School of Mathematical Sciences, Fudan University, Shanghai 200433, China*; orcid.org/0009-0006-7174-2097

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jctc.4c01530

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

The authors thank Jianfeng Lu and Zhe Wang for their helpful discussions and valuable insights during the course of this work. This research was supported by the National Natural Science Foundation of China (NSFC) under grant number 12271109, the National Key R&D Program of China under grant 2021YFA1003305, the Science and Technology

Commission of Shanghai Municipality (STCSM) under grant numbers 22TQ017 and 24DP2600100, and the Shanghai Institute for Mathematics and Interdisciplinary Sciences (SIMIS) under grant number SIMIS-ID-2024-(CN). The authors are also grateful for the resources and facilities provided by NSFC, the National Key R&D Program of China, STCSM, and SIMIS, which were essential for the completion of this work.

REFERENCES

(1) White, S. R.; Martin, R. L. Ab initio quantum chemistry using the density matrix renormalization group. *J. Chem. Phys.* **1999**, *110*, 4127−4130.

(2) Chan, G. K.-L.; Head-Gordon, M. Highly correlated calculations with a polynomial cost algorithm: A study of the density matrix renormalization group. *J. Chem. Phys.* **2002**, *116*, 4462−4476.

(3) Chan, G. K.-L.; Sharma, S. The Density Matrix Renormalization Group in Quantum Chemistry. *Annu. Rev. Phys. Chem.* **2011**, *62*, 465−481.

(4) Schollwöck, U. The density-matrix renormalization group in the age of matrix product states. *Ann. Phys.* **2011**, *326*, 96−192.

(5) Keller, S.; Dolfi, M.; Troyer, M.; Reiher, M. An efficient matrix product operator representation of the quantum chemical Hamiltonian. *J. Chem. Phys.* **2015**, *143*, 244118.

(6) Booth, G. H.; Thom, A. J. W.; Alavi, A. Fermion Monte Carlo without fixed nodes: A game of life, death, and annihilation in Slater determinant space. *J. Chem. Phys.* **2009**, *131*, 054106.

(7) Ghanem, K.; Lozovoi, A. Y.; Alavi, A. Unbiasing the initiator approximation in full configuration interaction quantum Monte Carlo. *J. Chem. Phys.* **2019**, *151*, 224108.

(8) Blunt, N. S.; Thom, A. J. W.; Scott, C. J. C. Preconditioning and Perturbative Estimators in Full Configuration Interaction Quantum Monte Carlo. *J. Chem. Theory Comput.* **2019**, *15*, 3537−3551.

(9) Cleland, D.; Booth, G. H.; Alavi, A. Communications: Survival of the fittest: Accelerating convergence in full configuration-interaction quantum Monte Carlo. *J. Chem. Phys.* **2010**, *132*, 041103.

(10) Yang, M.; Pahl, E.; Brand, J. Improved walker population control for full configuration interaction quantum Monte Carlo. *J. Chem. Phys.* **2020**, *153*, 174103.

(11) Eriksen, J. J. The Shape of Full Configuration Interaction to Come. *J. Phys. Chem. Lett.* **2021**, *12*, 418−432.

(12) Huron, B.; Malrieu, J.; Rancurel, P. Iterative perturbation calculations of ground and excited state energies from multiconfigurational zeroth-order wavefunctions. *J. Chem. Phys.* **1973**, *58*, 5745−5759.

(13) Tubman, N. M.; Lee, J.; Takeshita, T. Y.; Head-Gordon, M.; Whaley, K. B. A deterministic alternative to the full configuration interaction quantum Monte Carlo method. *J. Chem. Phys.* **2016**, *145*, 044112.

(14) Holmes, A. A.; Tubman, N. M.; Umrigar, C. Heat-bath configuration interaction: An efficient selected configuration interaction algorithm inspired by heat-bath sampling. *J. Chem. Theory Comput.* **2016**, *12*, 3674−3680.

(15) Sharma, S.; Holmes, A. A.; Jeanmairet, G.; Alavi, A.; Umrigar, C. J. Semistochastic heat-bath configuration interaction method: Selected configuration interaction with semistochastic perturbation theory. *J. Chem. Theory Comput.* **2017**, *13*, 1595−1604.

(16) Li, J.; Otten, M.; Holmes, A. A.; Sharma, S.; Umrigar, C. J. Fast Semistochastic Heat-Bath Configuration Interaction. *J. Chem. Phys.* **2018**, *149*, 214110.

(17) Wang, Z.; Li, Y.; Lu, J. Coordinate Descent Full Configuration Interaction. *J. Chem. Theory Comput.* **2019**, *15*, 3558−3569.

(18) Greene, S. M.; Webber, R. J.; Weare, J.; Berkelbach, T. C. Improved Fast Randomized Iteration Approach to Full Configuration Interaction. *J. Chem. Theory Comput.* **2020**, *16*, 5572−5585.

(19) Goings, J. J.; Hu, H.; Yang, C.; Li, X. Reinforcement Learning Configuration Interaction. *J. Chem. Theory Comput.* **2021**, *17*, 5482−5491.

(20) Li, Y.; Lu, J.; Wang, Z. Coordinate-Wise Descent Methods for Leading Eigenvalue Problem. *SIAM J. Sci. Comput.* **2019**, *41*, A2681−A2716.

(21) Dhillon, I. S.; Parlett, B. N.; Vömel, C. The design and implementation of the MRRR algorithm. *ACM Trans. Math. Softw.* **2006**, *32*, 533−560.

(22) Li, Y.; Lu, J. Optimal Orbital Selection for Full Configuration Interaction (OptOrbFCI): Pursuing the Basis Set Limit under a Budget. *J. Chem. Theory Comput.* **2020**, *16*, 6207−6221.

(23) *A high-performance, concurrent hash table.* https://github.com/efficient/libcuckoo. (accessed 7 May 2024).

(24) Turney, J. M.; et al. Psi4: an open-source ab initio electronic structure program. *Wiley Interdiscip. Rev.: Comput. Mol. Sci* **2012**, *2*, 556−565.

(25) *C++ implemented Coordinate Descent Full Configuration Interaction (CDFCI) package for quantum chemistry calculation.* https://github.com/quan-tum/CDFCI. (accessed 21 June 2024).

(26) *Arrow - Fast Semistochastic Heat Bath Configuration Interaction Solver (SHCI).* https://github.com/QMC-Cornell/shci. (accessed 21 June 2024).

(27) *Efficient parallel quantum chemistry DMRG in MPO formalism.* https://github.com/block-hczhai/block2-preview. (accessed 21 June 2024).

(28) *Standalone NECI codebase designed for FCIQMC and other stochastic quantum chemistry methods.* https://github.com/quan-tum/CDFCI. (accessed 21 June 2024).

(29) Larsson, H. R.; Zhai, H.; Umrigar, C. J.; Chan, G. K.-L. The Chromium Dimer: Closing a Chapter of Quantum Chemistry. *J. Am. Chem. Soc.* **2022**, *144*, 15932−15937.

(30) Olivares-Amaya, R.; Hu, W.; Nakatani, N.; Sharma, S.; Yang, J.; Chan, G. K.-L. The ab-initio density matrix renormalization group in practice. *J. Chem. Phys.* **2015**, *142*, 034102.

(31) Kreplin, D. A. *Multiconfiguration Self-consistent Field Methods for Large Molecules*, Ph.D. thesis; Universität Stuttgart, 2020.

(32) Smith, J. E. T.; Mussard, B.; Holmes, A. A.; Sharma, S. Cheap and Near Exact CASSCF with Large Active Spaces. *J. Chem. Theory Comput.* **2017**, *13*, 5468−5478.

(33) Li, J.; Yao, Y.; Holmes, A. A.; Otten, M.; Sun, Q.; Sharma, S.; Umrigar, C. J. Accurate many-body electronic structure near the basis set limit: Application to the chromium dimer. *Phys. Rev. Res.* **2020**, *2*, 012015.

(34) Blunt, N. S.; Smart, S. D.; Booth, G. H.; Alavi, A. An excited-state approach within full configuration interaction quantum Monte Carlo. *J. Chem. Phys.* **2015**, *143*, 134117.

(35) Holmes, A. A.; Umrigar, C. J.; Sharma, S. Excited states using semistochastic heat-bath configuration interaction. *J. Chem. Phys.* **2017**, *147*, 164111.

(36) Schriber, J. B.; Evangelista, F. A. Adaptive configuration interaction for computing challenging electronic excited states with tunable accuracy. *J. Chem. Theory Comput.* **2017**, *13*, 5354−5366.

(37) Gao, W.; Li, Y.; Lu, B. Triangularized orthogonalization-free method for solving extreme eigenvalue problems. *J. Sci. Comput.* **2022**, *93*, 63.

(38) Gao, W.; Li, Y.; Shen, H. Weighted trace-penalty minimization for full configuration interaction. *SIAM J. Sci. Comput.* **2024**, *46*, A179−A203.

(39) Wang, Z.; Zhang, Z.; Lu, J.; Li, Y. Coordinate Descent Full Configuration Interaction for Excited States. *J. Chem. Theory Comput.* **2023**, *19*, 7731−7739.

https://doi.org/10.1021/acs.jctc.4c01530
J. Chem. Theory Comput. 2025, 21, 2325−2337