Quantum Computation as a Game: Graph State Searching Protocols for Quantum Chemistry Simulations

Eegan Ram

Under the direction of

Dr. Di Luo, Zhuo Chen, Dr. Rumen Dangovski, and Professor Marin Soljačić
Postdoc, Ph.D Candidate, Ph.D, Professor of Physics
Massachusetts Institute of Technology

August 1, 2023

Abstract

In order to demonstrate the potential to minimize quantum resource usage by presenting quantum computing tasks as optimization "games," the quantum chemistry problem of finding the ground state energies of atoms at given bond lengths in hydrogen and water molecules was solved using that method. To determine the ground state energy of a system at a given bond length, the ground state wavefunction was found using the technique of searching through graph states and local unitaries. In combination, they represent a Clifford state that provides a reasonable approximation of the ground state for a given bonding distance at quadratic rather than exponential cost. While hydrogen could be solved using a brute force search, various algorithms were employed to search through the graph states for water molecules including simulated annealing and an implementation of Monte-Carlo Tree Search with a policy and value network made possible by reframing the graph search game as a tree search problem. Both methods were able to identify energies closer to the exact ground state energy than the Hartree-Fock approximation and converge to the exact value at large bond lengths.

Summary

To demonstrate the potential of presenting quantum computing problems as "games," the example quantum chemistry problem of finding the bonding distances between atoms in hydrogen and water molecules was framed as a game. In order to determine the lowest possible energy that a molecule could have at a given bond length, the wavefunction that describes that lowest-energy state was found using the technique of searching through different configurations of a graph that represents the entanglement between electrons in the molecule. Different methods that could search through all possible connections on that graph were used, including simulated annealing, which involves making repeated random changes to the graph, and Monte-Carlo Tree Search, which involves searching through a tree of possible configurations of the graph guided by a neural network. Both methods were able to identify lower energy states than existing approximations, while using fewer resources to complete the computation.

1 Introduction

With many recent developments in the field of quantum computing promising to provide significant speed benefits to algorithms too time consuming for traditional computers to accomplish, many scientists believe that quantum advantage is now about to become a reality [1]. However, despite these new developments, the allocation of quantum resources remains expensive. Therefore, it is advantageous to find alternate methods of minimizing the use of those resources as they pertain to common quantum computing tasks.

One such task that quantum computers are capable of solving pertains to a classic quantum chemistry problem: finding the lowest energy states of various entangled systems in order to find the bonding distances between atoms in complex molecules. For molecules with many electrons, finding the bonding distances between atoms can be computationally intensive, as electrons in the molecule can be in a variety of possible entangled states with one another, influencing the energy of the entire system. Thus, the task consists of calculating the ground state energy for a given bonding distance in a system of electrons.

Several methods exist for tackling this problem, including algorithms that run on classical and quantum computers. One commonly used classical technique is the Hartree-Fock method, which works by approximating the wavefunction of the system as a single (Slater) determinant. It therefore essentially assumes that each electron's motion can be described by a wavefunction that does not depend on the instantaneous motions of other electrons in the system and completely neglects relativistic effects [2]. The method is relatively cheap computationally but does lead to inaccurate calculations that worsen as bond length increases.

Other methods using quantum computers do exist, most notably quantum adiabatic evolution, phase estimation, and the rodeo algorithm. Of those, adiabatic evolution and phase estimation are extremely cost prohibitive computationally in near term quantum computers [3]. In comparison to those, the rodeo algorithm is exponentially faster but utilizes auxillary qubits that are coupled to the system, which increases its quantum resource usage beyond what would otherwise be required for simulation [3].

To create a method that is less cost prohibitive and minimizes its quantum resource usage, we utilize the technique of searching through graph states, which represent multi-qubit entangled states, along with local unitaries in order to find the ground state wavefunctions that represent systems of electrons in certain molecules. This search process is framed as a "quantum game" of sorts that motivates the usage of common algorithms to complete the search. An agent that takes energy measurements of the quantum state plays this game by choosing new graph states and local unitaries to search using various algorithms. The agent's goal is to find the appropriate graph state and local unitary to minimize the energy of the system, maximizing its reward. As the set of game states proved too large for brute-force search algorithms to be used on molecules larger than hydrogen, simulated annealing and a unique implementation of Monte-Carlo Tree Search (MCTS) were used in order to search the game for the state with the lowest energy.

While these algorithms have been used for the purpose of finding ground states in the past, the method with which the search space is constructed differs from other work. For instance, simulated annealing has been used in classical computing environments to search through existing sets of measurement data to find spin-glass ground states [4]. Meanwhile, Monte-Carlo Tree Search has been utilized in similar hybrid quantum-classical frameworks for the purpose of searching through quantum circuit spaces directly [5]. However, this method utilizes a quantum circuit approach rather than our method's graph states, the latter of which are compact and memory efficient, opening up new possibilities for the application of these established algorithms.

2 Background

In quantum information science, graph states are one of the best methods of representation of entangled states due to their tendency to grow reasonably in size with the size of the system [6]. In our methodology, they also serve as algorithmic resources for our one-way quantum computer. A graph state is defined as a set of n vertices, corresponding to the n

qubits of an entangled state. They are commonly represented as adjacency matrices.[1]

Graph states are particularly useful for simulating easily-scalable Clifford states, which are states that can be represented by Clifford operators[2] in combination with computational basis states [7]. While Clifford states represent only a subset of all possible quantum states, they are particularly useful as they are cost-quadratic with the number of qubits used in the system, rather than the cost-exponential situation of representing all possible states. However, not all Clifford states can be directly calculated by calculating the wavefunction directly from a graph state, as additional Clifford operators can be applied. The way to complete the set of stabilizer states that can be calculated from graph states is by incorporating local unitary transformations.

Local unitary transformations consist of a sequence of single qubit Clifford gates that do not change the entanglement of a given state. There are 23 such transformations, excluding the identity transformation, that can be applied to change the state to distinct values. They have a variety of different representations but all be represented as a series of Hadamard and S gates.[3] As such, to search through all possible Clifford states, one must search through all permutations of 23 possible local unitary transformations on each qubit and all possible graph states that can be represented by that system of qubits.

To search through these, the simulated annealing and Monte-Carlo Tree Search (MCTS) algorithms were chosen. Simulated annealing is already a commonly used algorithm in the field of quantum chemistry. Its original purpose is to solve energy optimization problems [8], making it a reasonable fit for the situation being handled here. However, it does have problems with wasted computation in biased distributions, and occasionally returns local minima due to its built-in randomness.

MCTS is a more novel method as it applies to this situation. It is perhaps most famous

[1] An adjacency matrix is an $n \times n$ matrix that represents an n vertex graph, where a 1 in position i, j means that vertices i and j are connected by an edge, while a zero indicates they are disconnected.

[2] The clifford operators are: X, Y, Z, I, H, S, $S\dagger$, $\sqrt{X}$, $\sqrt{X}\dagger$, CX, CY, CZ, and Swap.

[3] $\mathrm{H} = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}; S = \begin{bmatrix}1 & 0\\ 0 & i\end{bmatrix}$

for its use in decision based games, such as Go, and various strategy video games. It has additionally been used in other computing optimization tasks such as matrix multiplication, however, these optimizations were created by formulating a game out of the matrix-multiplication calculation, much like the approach we take with finding ground states [9]. In many cases, such as that of the Go solving algorithm AlphaGo, it is coupled with value and policy neural networks that advise its predictions of the best moves and the value of a given game state, respectively [10]. As such, the tree search looks for possibilities guided by the policy network, while the value network assesses the probability of winning from a certain state. In our game, the set of game states can be represented as a tree with which MCTS can be used to search.

3 Methods

3.1 Constructing The Quantum Game

The game consists of tasking an agent taking measurements from a quantum computer to find the lowest energy state of a set of qubits, representing electrons in a molecule. This is done by searching through graph states and local unitaries with some algorithm.

A graph can be converted directly into its corresponding quantum graph state using:

$$|G\rangle = \prod_{(a,b)\in E} U^{\{a,b\}} |+\rangle^{\otimes V}$$

where V and E are the set of vertices and edges on the graph, $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, and $U^{\{a,b\}}$ is the controlled-Z gate between the two qubits corresponded to by vertices a and b [6]. However, it is more efficient for both memory and calculation purposes to make use of the graph state's stabilizer state[4] which can be found using the formula:

$$K_j = \sigma_x^{(j)} \prod_{k=1}^{n} (\sigma_z^{(k)})^{\theta_{kj}}$$

Where $\sigma_x^{(i)}$ and $\sigma_z^{(i)}$ are the Pauli operators, matrices, acting on the ith position of the

[4] A stabilizer state is a set of Pauli operators, such as X, Y, and Z, that give an eigenvalue of 1, or do not change the state, when multiplied by their corresponding state. Each stabilizer state has a unique state that it stabilizes, though a single state can have multiple stabilizer states that represent it.

tensor product, and θ is the adjacency matrix of the graph [11]. This allows for a relatively simple conversion from a graph state to a stabilizer state that represents its corresponding wafefunction and energy calculation. Local unitary transformations can then be directly applied to these stabilizer states, expanding the possible search area to all Clifford states. Thus, the lowest energy wavefunction can be found by testing possible graph states through modifying adjacency matrices, applying local unitaries, and calculating stabilizer states.

For algorithms that utilize the graph state directly, then, this is the final construction of the game and it is up to the agent to decide how to choose graphs in order to achieve the lowest energy. However, for reinforcement learning methods like MCTS, it is beneficial to frame the game as a tree search problem instead. The structure can be seen in figure 1.

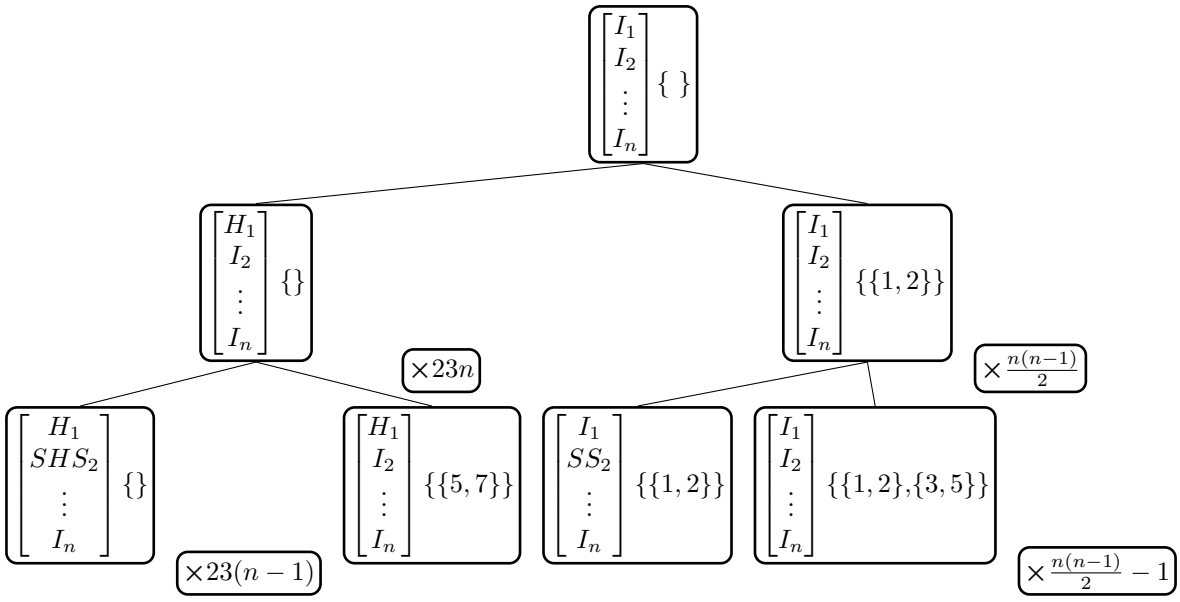

Figure 1: Example Tree of 3 Layers. The column vector represents the local unitaries being applied to each qubit while the set is the set of connected edges on the graph.

At the starting node, no vertices on the graph are connected and no local unitaries are applied to any qubit. There are then 23 options to apply a local unitary to each of n qubits, leaving $23 \cdot n$ options for local unitaries, in addition to $\frac{n(n-1)}{2}$ options to connect two vertices on the graph. To avoid loops in the tree, once a local unitary is added or two vertices connected, that qubit's local unitary cannot be overwritten or those two vertices

disconnected.

It is worth noting that while the tree is highly branched, it is not particularly deep. For example, water, which has 10 electrons, would have 275 branches, 230 of which are local unitary transformations and 45 of which are changes to the graph. However, the depth of the tree is only $\frac{n(n-1)}{2} + n$, or 55 for a 10 electron molecule. This is due to the characteristic that a qubit's local unitary cannot be overwritten, therefore only n local unitaries can be set.

3.2 Simulated Annealing

One technique used for optimizing the graph state picking algorithm was simulated annealing. Simulated annealing is an optimization technique inspired by annealing in metallurgy as studied in statistical mechanics [8]. Our implementation of the algorithm is detailed in algorithm 1.

The temperature T decreases over time, in a number of steps equal to the number of epochs from 0.1 to 0.01. These temperature parameters were chosen empirically. As such, changes that increase the energy are more likely to be permitted earlier rather than later in the process. Therefore, the algorithm attempts to find the lowest energy state, though it does permit an occasional raising of the energy particularly during the start of the algorithm. This process was then repeated for a variety of values of d in order to generate a graph of bonding distance vs. energy.

An additional improvement was made by implementing transfer learning. This is done by restarting the algorithm but by starting with the state finished by a previous run of the algorithm. Transfer learning helps to improve stochastic algorithms as it makes it more likely for the partially random algorithm to escape local minima should it have settled on one in the first iteration of the algorithm. It therefore exploits the knowledge of existing minima gained from the previous iteration of the algorithm to improve its performance on the next transfer.

for *each distance* **do**
  Start with a completely disconnected graph.
  Randomly and uniformly choose two nodes to toggle their connection in sequence.
  for *#epochs* **do**
    Calculate the energy of this new state.
    if $\exp\left(\frac{-\Delta E}{T}\right) > x$ *where* x *is chosen uniformly at random from* $[0, 1)$ **then**
      accept this as the new state.
    end
    if *the new state has the new lowest energy* **then**
      store its energy as the minimum.
    end
    Randomly and uniformly choose a qubit and a local unitary to apply.
    if $\exp\left(\frac{-\Delta E}{T}\right) > x$ **then**
      accept this as the new state.
    end
    if *the new state has the new lowest energy* **then**
      store its energy as the minimum.
    end
  end
end

Algorithm 1: Simulated Annealing Algorithm for Searching Through Graph States

3.3 Monte-Carlo Tree Search

Monte-Carlo Tree Search (MCTS) is a heuristic search algorithm most famously implemented in order to master the game of Go in the program AlphaGo Zero [10]. Our implementation is quite similar to the implementation detailed in the AlphaGo Zero paper. As in AlphaGo Zero, the MCTS is accompanied by a policy neural network that predicts move selection, and a value network that predicts the value of a given state. The policy and value network consist of a common dense layer, another dense layer for each, and 20 hidden layers. The MCTS algorithm itself has four main phases:

1. **Selection**: A leaf node[5] is selected by finding the node with the maximum value of the action probability, given by
$$U(s,a) + Q(s,a) = c_{puct} P(s,a) \frac{\sqrt{1 + n_{visits}}}{1 + n_{children}} + Q(s,a)$$
where c_{puct} is the exploration constant, $P(s,a)$ is the prior probability of selecting that node, and $Q(s,a)$ is the mean action value. In our implementation, illegal actions, such as reassigning a qubit or disconnecting an edge that would create loops are taken out of the search space by assigning negative infinite action values to ensure they will never be selected.

2. **Expansion**: The leaf node is expanded by using the value network to estimate the value of its children. The policy network helps determine which leaves to explore.

3. **Simulation**: the search algorithm selects random actions until the game reaches its final state. In our case, this happens at the end of the tree when all nodes are connected and each qubit has an applied local unitary.

4. **Backpropagation**: The action values are updated to the mean weighting values in a backwards pass. It is important to note that our reward value diverges from the AlphaGo approach as it is given by the negative of the minimum energy along the entire branch, rather than the energy at the end of the tree. This is because it is meaningless as to whether or not the ground state occurs with fully connected nodes, and the algorithm should not be incentivized to skip through the ground state.

The policy network is then trained based on the experience gained from the tree search process by giving it a mini batch of 32 per worker, in an identical fashion to the AlphaGo Zero algorithm. The unique reward mechanism is necessary because unlike the game of Go or other traditional applications of Monte-Carlo Tree Search, there is no well defined "win state" to define the end of the simulation step. However, as the tree of possible graph states and local unitaries has a reasonably sized depth, it is possible to search until the end of

[5]A node that does not yet have explored child nodes.

the graph, which implies that all vertices are connected to all others with edges, and all 10 qubits have applied local unitaries.

We chose to implement a similar algorithm as the game of Go shares several characteristics with our game tree, most notably its heavily branched structure, but also its tendency to have "completed states" at unusual locations. For instance, resignation is a common way for a game of Go to end, and can occur before all possible pieces can be placed, which is similar to how the ground state energy most likely does not occur at the point where all nodes on the graph are connected, and all local unitaries are applied, which is the end of the tree. We therefore assumed that its similar characteristics and game tree would make its specific implementation an ideal candidate algorithm for the situation.

Although "illegal moves" that rewrite unitaries or disconnect edges on the graph were initially allowed, due to poor performance with that method, the illegal action policy was introduced to discourage such moves.

4 Results

4.1 Simulated Annealing

Upon implementing the Simulated Annealing Algorithm for the hydrogen atom, the result in figure 2 was realized.

The result matched the brute-force result, which was possible to implement for hydrogen due to its small electron count. This correspondence confirmed that the algorithm was able to find the best possible energy state that a quantum computer could calculate. The quantum computer is unable to do better than the Hartree-Fock approximation until around 1.5 bond lengths, where it suddenly begins to converge to the exact distance. This is because our graph state search only covers a subset of quantum states that can be represented, Clifford states. The general result is not a Clifford state which is why the quantum computer's result does not match the exact result. This is more of an issue for smaller molecules where it

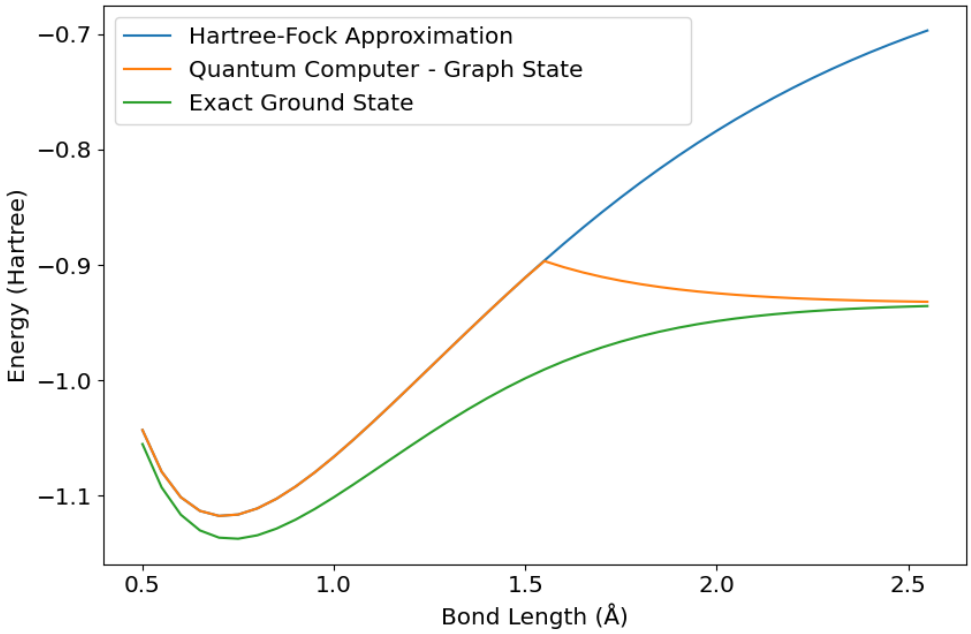

Figure 2: H2 Bond Length vs. Energy With Simulated Annealing

is possible to calculate the actual quantum state classically, but the ease of using Clifford states becomes more useful when the molecules become larger. As Clifford states only have a quadratic cost while the set of all states has an exponential cost, the exact ground state may become near impossible to resolve due to its computational expense, while the graph state search algorithm will become an acceptable result that is better than the HF approximation.

The simulated annealing algorithm was then tested on H2O as seen in figure 3.

The Hartree-Fock approximation is not useful at longer bond lengths which is why the calculation was stopped around 2.5 Å. As can be seen, the simulated annealing algorithm quite clearly performs better in this situation in finding the ground state energy at larger bond lengths, though it does have some roughness in certain locations motivating the usage of a different algorithm.

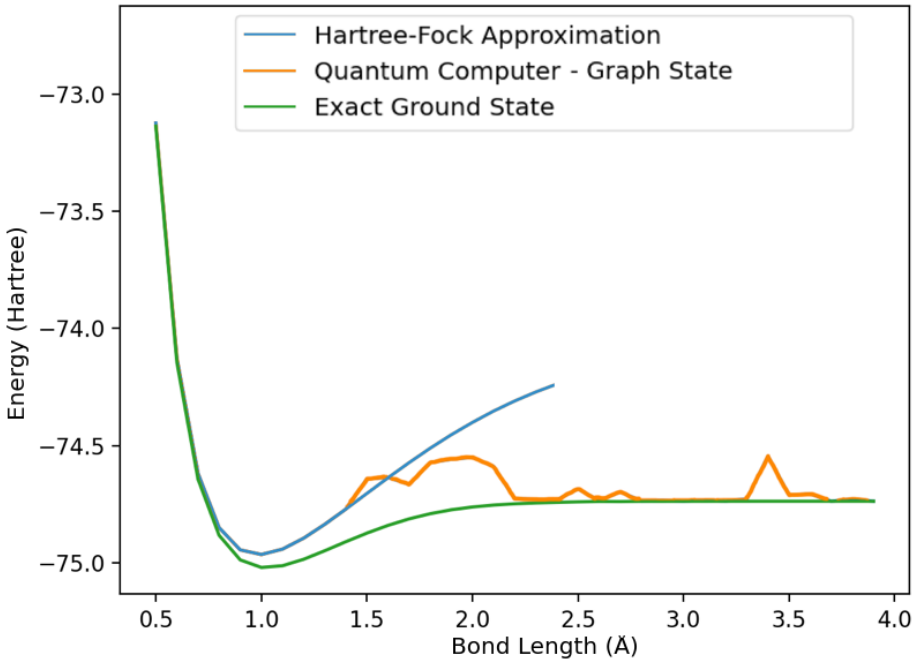

Figure 3: H2O Bond Length (Å) vs Energy (Hartree) with Simulated Annealing

4.2 Monte-Carlo Tree Search

Upon testing Monte-Carlo Tree Search for finding the optimal distance between hydrogen atoms, a graph identical to figure 2 was obtained after 150 training iterations.

Scaling up to water, the MCTS took 500 training episodes to achieve a reasonable result, as shown in the energy graph in figure 4. Further training may have further improved the result, but the algorithm does have to be trained on each individual bond length's tree for it to be viable.

The graph demonstrates a similar curve in the range between 1.5 and 2.5 Å when compared to simulated annealing, but proved to run closer to the exact ground state, therefore achieving a marginally higher accuracy. However, it does contain a number of spikes that are noticably more sparce when using simulated annealing. However, in general, the algorithm matched or outperformed the simulated annealing technique most of the time.

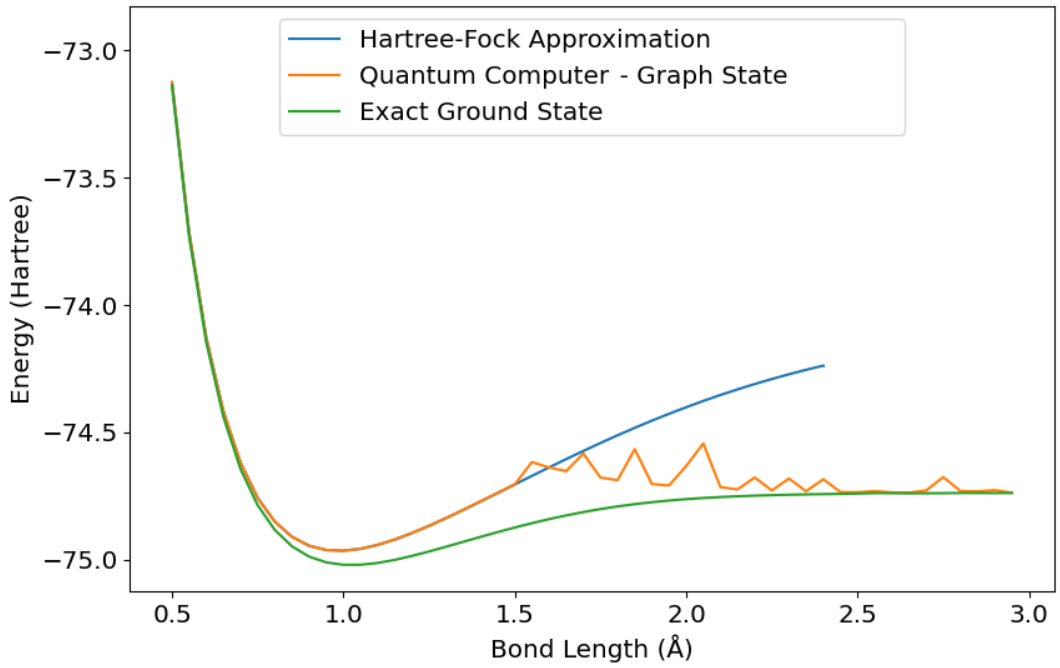

Figure 4: Bond Length vs. Energy for H2O with Monte-Carlo Tree Search

5 Discussion

The success of the simulated annealing algorithm in particular in often finding the Clifford states that are the closest to the exact ground states is not necessarily surprising, as the algorithm itself is designed to mimic annealing in statistical mechanics, which is not only physically related to molecular bonding but also shares many of its random characteristics.

Despite the power of the Monte-Carlo Tree Search algorithm with deep learning, its difficulty in reconciling with the ultimately less patterned tree structure that the game represents likely hinders its progress at finding the optimal ground state. Comparing the tree structure of the Clifford search to other implementations of MCTS, such as in AlphaGo, a single move on the board in Go usually only incrementally changes its state towards a certain outcome. However, with the quantum chemistry implementation, merely applying another local unitary, representing a single move down a branch of the tree, could significantly change the energy of the state.

However, framing the problem as a tree search in this way does represent a significant departure from other methods of combining reinforcement learning with more traditional techniques, such as in feedback gradient-ascent pulse engineering (feedback-GRAPE) which utilizes concepts from model-free reinforcement learning in order to optimize state preparation and stabilization of cavity quantum electrodynamics scenarios [12], which are somewhat similar to the quantum computer optimization work presented here.

It is worth repeating that the method of searching through Clifford states is still unable to resolve the exact ground state energy due to Clifford states only representing a smaller subsection of all states that can be held. However, as the method is purely attempting to bridge the gap between existing algorithms, it does appear to fulfill this criterion. Future research could focus on potentially further optimizing Monte-Carlo tree search or expanding this methodology to larger molecules, which may lead to even greater benefits from applying this method.

6 Conclusion

Through the course of attempting to design and optimize algorithms to solve a quantum game that finds the bonding distances between atoms by finding the ground states of systems at given bonding distances, we have found that utilizing algorithms such as simulated annealing and Monte-Carlo Tree Search for this task is a viable method for performing this calculation. Overall, while searching through Clifford states generated from graph states cannot provide the exact ground state, we have found that instituting a gamified system for finding states that well approximate those ground states is a reasonable way to solve this problem that simultaneously reduces quantum resource usage and computational complexity when compared to existing exact methods while being more accurate than the Hartree-Fock approximation.

7 Acknowledgments

Thank you to my mentors, Dr. Di Luo, Mr. Rumen Dangovski, and Dr. Zhuo Chen, as well as my faculty supervisor and Principal Investigator of our lab, Professor of Physics Marin Soljačić. I would also like to thank the Center for Excellence in Education and its sponsors, its Research Science Institute, the Massachusetts Institute of Technology, my tutor, Mr. Shuvom Sadhuka, and reviewer Mr. Max Bee-Lindgren. Furthermore, thank you to my parents, RSI counselor, friends, other RSI peers in the same lab, Rayhan Divo Tanudjaja and Connie Guo, for supporting me during this research period. I would also like to provide thanks to Dr. Michael Ke Zhang for helping me enter the intersection between AI and Physics. Lastly, I would also like to thank my teachers Mrs. Wait, Mr. Guisti, and Mrs. Hong for lending me the necessary skills to help me get here.

References

[1] W. Huggins, P. Patil, B. Mitchell, et al. Towards quantum machine learning with tensor networks. *Quantum Science and Technology*, 4(2):p. 024,001, 2019. doi:10.1088/2058-9565/aaea94.

[2] C. D. Sherrill. An introduction to hartree-fock molecular orbital theory. *School of Chemistry and Biochemistry Georgia Institute of Technology*, 2000.

[3] K. Choi, D. Lee, J. Bonitati, et al. Rodeo algorithm for quantum computing. *Physical Review Letters*, 127(4), 2021. doi:10.1103/physrevlett.127.040505.

[4] W. Wang, J. Machta, and H. G. Katzgraber. Comparing monte carlo methods for finding ground states of ising spin glasses: Population annealing, simulated annealing, and parallel tempering. *Physical Review E*, 92(1), 2015. doi:10.1103/physreve.92.013303.

[5] J. Yao, H. Li, M. Bukov, et al. Monte carlo tree search based hybrid optimization of variational quantum circuits, 2022. `2203.16707`.

[6] M. Hein, W. Dür, J. Eisert, et al. Entanglement in graph states and its applications. 2006. `quant-ph/0602096`.

[7] S. Aaronson and D. Gottesman. Improved simulation of stabilizer circuits. *Physical Review A*, 70(5), 2004. doi:10.1103/physreva.70.052328.

[8] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):pp. 671–680, 1983. doi:10.1126/science.220.4598.671.

[9] A. Fawzi, M. Balog, A. Huang, et al. Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):pp. 47–53, 2022.

[10] D. Silver, J. Schrittwieser, K. Simonyan, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):pp. 354–359, 2017.

[11] M. V. den Nest, J. Dehaene, and B. D. Moor. Graphical description of the action of local clifford transformations on graph states. *Physical Review A*, 69(2), 2004. doi:10.1103/physreva.69.022316.

[12] R. Porotti, V. Peano, and F. Marquardt. Gradient ascent pulse engineering with feedback. 2023. `2203.04271`.