Reinforcement Learning Agents With and Without Access to Quantum Computation

Authors and Affiliations

Geordie Rose, Snowdrop Quantum Applications Corporation

Contribution Type: New Application

Project Abstract: This project explores under what conditions reinforcement learning (RL) agents with access to quantum computation (QC) can be *categorically* superior to agents with only access to classical computation. To this end, we investigate the problem of training RL agents to design quantum systems, such as molecules, materials, and chemical reactions, with desirable properties. 'Desirable properties' are encoded in the reward signal as functions of the system's observables, making quantum simulation, one of the most promising use cases for QC, required to extract reward. In this Phase 1 submission, we focus on the case where computing reward requires evaluating functions where there are current claims of quantum advantage. Agents without QC access must approximate reward using classical methods, which we treat as learning from corrupted reward signals. To make these ideas concrete, we introduce and analyze a two-player zero-sum game called *Tangled*, where evaluating reward is a domain where quantum advantage has been claimed.

August 2, 2025

A. Problem Statement & Scope

Problem Statement

A Reinforcement Learning (RL) agent attempts to develop a policy - a mapping of states to actions - that maximizes cumulative reward over time [1]. It has been argued that the RL paradigm is sufficient to explain the evolution of intelligence and associated properties of mind in all biological agents, including humans, and it is the standard framework within which current attempts to build artificial general intelligence and superintelligence sit [2, 3, 4].

There exist problems where quantum computation (QC) has a fundamental advantage over classical computation [5, 6, 7, 8, 9, 10, 11]. It has been shown that giving RL agents access to QC can provide a *categorical* advantage in goal-seeking capability – that is, a separation not only of degree but of kind – over agents without such access [12].

In this project, we seek to discover goal types where categorical advantage might be had for agents with access to near-term QC systems, and if such exist, implement agents with this advantage. Because we seek categorical advantage, we need to find a strategy that attacks a fundamental aspect of the RL paradigm. A common approach is to replace neural nets with quantum circuits [13, 14, 15], but we instead propose to continue to use neural nets as general function approximators and instead focus our attention on RL's reliance on accurate computation of reward. It is known that corrupting the reward signal can have devastating effects on an agent's ability to learn [16]. To attempt to exploit this feature, we start by examining problems where reward is most naturally computed via simulation of a quantum system. Simulating quantum systems is a problem category where QC has fundamental advantages over classical computation [17]. If we want an agent to achieve goals that require simulating quantum systems, giving access to QC might provide the categorical advantage we seek. This motivates the following problem statement.

Problem Statement: In the case where reward is the output of a quantum simulation, we aim to (a) discover what conditions are sufficient for RL agents with access to QC to be categorically superior to agents without such access, and (b) solve a set of industrially important challenge problems requiring design of quantum systems using such agents.

Phase 1 Scope

We initially narrow our scope to examine the special case where the quantum simulation generating reward is a domain where *quantum advantage* has already been claimed. Quantum advantage refers to a situation where a quantum computer can solve a problem that is either impossible or impractical for classical computers to solve within a reasonable timeframe or using feasible resources. There are currently three domains in which quantum advantage claims exist: random circuit sampling [18], boson sampling [19], and simulating quenches through quantum phase transitions in magnets [20], although none of these are universally accepted by the scientific community.

Specifically, we restrict our attention to the case where the agent-environment interaction is a Markov Decision Process (MDP), and the reward returned from an environment containing state S_t is $R_t = F(S_t)$, where F is any function where a current claim of quantum advantage exists (see Fig. 1 and Fig. 3.1 in [1]).

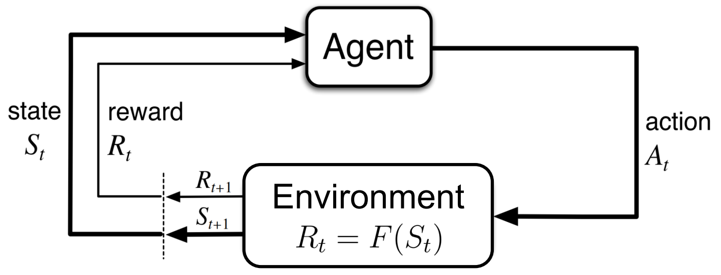


Figure 1: The agent–environment interaction in a Markov Decision Process, restricted to where reward $R_t = F(S_t)$, where F is any function where a current claim of quantum advantage exists. This is a special case of the more general case where F is the result of any quantum simulation.

Scope: In this Phase 1 report, we examine a concrete example of this type of problem, which we formulate as a fully observed zero-sum two-player game with terminal states we call *Tangled*. A terminal state S_T is a system state where the game being played has ended, and a result must be obtained to determine the outcome. Familiar games in this category include Chess, Checkers, and Go. The reward associated with a Tangled terminal state is $R_T = F(S_T) \in \{+1, -1, 0\}$, denoting the reward obtained from a win, loss, or draw, and F is a function where there is a current claim of quantum advantage.

B. Impact on the Problem Area

The impact of recent advances in artificial intelligence (AI) has been profound. There are few areas of human endeavor where AI has not already had impact. As AI improves towards artificial general intelligence (AGI) and superintelligence, and moves from digital to embodied form, it is likely that eventually AI will be viewed as one of the most important transformative technologies ever developed [21].

The resources currently being spent on advancing AI are remarkable. Technology companies plan to spend over \$300 billion in capital expenditures in 2025, up from \$230 billion in 2024 [22], while Goldman Sachs research indicates that corporations and governments plan to spend around \$1 trillion on capital expenditures in the coming years to support the development and deployment of AI [23]. Governments worldwide are simultaneously racing to secure control over critical supply chains, including computational resources for data centers and material resources essential for embodied AI applications, including in defense applications where, for example, AI for drones has already significantly impacted the outcome of major conflicts [24, 25].

While there has been an explosion of development in building agents, this has almost exclusively been restricted to classical agents in purely classical environments. In the work described here, we aim to demonstrate that there exists an opportunity for QC to be seriously considered in the ongoing understanding of intelligence, agency, and the ultimate power of RL agents. We are particularly interested in high-value uses of RL agents in cases where giving these agents access to QC provides categorical advantage over agents without such access. A potential such use, and the one we focus on here, is where the goal of the agent is to *design quantum systems with desirable properties*.

The work presented here is a simplified version of the following general problem. Given a quantum system parametrized by some state space; an agent with a set of actions it can take to modify the system's state (such as changing the positions of nuclei in a molecule, adding an atom to a specific location, or modifying coupling strengths between spins); a desirable set of properties of the system (such as stability, binding affinities, chemical reaction rates, conductivity, etc.) that can be extracted via quantum simulation of this system; and a reward function that depends on those properties. If we have all these ingredients, we can train RL agents to design quantum systems with desirable properties using the techniques described here.

Currently, the range of quantum systems and reward functions where these ideas can be implemented – regimes in which quantum advantage has already been claimed – is limited. As technology advances, the regimes in which we can usefully perform quantum simulation with quantum computers should expand to encompass much more complex and interesting systems.

Agents capable of designing arbitrary quantum objects with specified properties would represent a transformative capability for multiple commercial sectors. Such a system could accelerate pharmaceutical development by enabling the rational design of drug molecules with optimized binding affinities, reduced side effects, and improved pharmacokinetic properties [26, 27], potentially reducing the typical 10-15 year drug development timeline and associated costs exceeding \$1 billion per approved compound [28]. In materials science, such agents could facilitate the discovery of novel catalysts for industrial processes [29], high-performance semiconductors for electronics applications [30], and energy storage materials with enhanced capacity and cycling stability [31]. The chemical industry could benefit from optimized reaction pathways and process conditions, leading to improved yields, reduced waste, and lower energy consumption [32]. Additionally, the capability to design quantum materials with tailored electronic, optical, or magnetic properties could enable breakthroughs in computing hardware, photovoltaic devices, and advanced sensors [33, 34]. The economic impact would likely be measured in hundreds of billions of dollars annually across these sectors.

More fundamentally, and perhaps even more interestingly (at least to us), agents with access to efficient and general quantum simulation, which presumably requires QC, gain the capability to model the world as it truly is. Nature appears to be quantum mechanical. If we want synthetic intelligences to be able to do science, we should endeavor to give them the tools they need to understand and predict how systems behave in the real world.

In addition to the possibility of demonstrating categorical advantage of agents with access to QC over those that are restricted to classical computation, we also hope that the current work will provide new sets of tools to the community with which to study the topic of quantum advantage itself. Instead of focusing on whether a quantum system computing a particular function F can do so with quantum advantage, we can focus on whether an agent with access to F can outperform classical agents without access to QC. While it might seem to introduce unnecessary complexity, thinking of whether a quantum computer can be used to train an agent to be better at playing an easily understood game (or more generally, design quantum systems with desirable properties) may provide a more interesting, valuable, and compelling testbed than just the basic function itself.

C. Quantum Advantage

In this Phase 1 submission we restrict our scope to considering only functions F (see Fig. 1) where reasonable claims of quantum advantage exist. Over time, each of these claims will be tested and some or all may be overturned with advancing classical algorithms and specialized approaches. It is our view that ultimately this is a one-sided battle, and while timescales remain uncertain, eventually QC systems will mature enough so that quantum advantage claims will transition from contested to universally agreed upon in the community.

If we assume quantum advantage for computing F as a necessary condition for what follows, there is still the question of how (or if) this translates to the ultimate power of agents trained with or without access to QC. We approach this in the following manner.

We treat the computation of reward $R_t = F(S_t)$ by quantum computing hardware as representing the true reward, and treat any approximate computation $\bar{R}_t = \bar{F}(S_t)$ using a classical approximation $\bar{F}$ as introducing corruption of the reward signal whenever $\bar{R}_t \neq R_t$. There are strong results known about the evaluation of agents in the presence of corrupted reward [16] that indicate the possibility of categorical advantage of RL agents with access to QC. Specifically, there are reward corruption rate thresholds over which no agent can learn to outperform an agent that acts purely randomly. If classical techniques cannot overcome these thresholds, then no agent trained using classical simulation of F can learn to do better than a random agent, and therefore agents that use the ground truth for F will be categorically superior.

A New Type of Game

While ultimately our objective is to explore the use of agents for designing high-value novel quantum systems, we begin by stripping away all of the complexities of the real world and focus on agents designed to play a new type of game.

Games provide well-defined environments with clear objectives that facilitate the evaluation and advancement of AI algorithms. The work of Shannon on chess programming in the 1950s established games as a natural domain for exploring computational intelligence [35], while the development of specialized game-playing systems like Deep Blue, which defeated world chess champion Garry Kasparov in 1997, demonstrated the potential for AI to exceed human performance in complex strategic tasks [36]. The introduction of Monte Carlo Tree Search algorithms revolutionized game AI and found applications beyond gaming [37], while the advent of deep reinforcement learning marked a paradigm shift with systems like AlphaGo achieving superhuman performance in Go through self-play and neural network function approximation [38]. Subsequently, multi-agent reinforcement learning in games such as Dota 2 and StarCraft II has advanced our understanding of complex strategic reasoning and emergent behaviors [39, 40], while the development of general game-playing agents has pushed toward more flexible and transferable AI systems. Gaming environments continue to serve as laboratories for testing new architectures, training methodologies, and theoretical frameworks.

The new kind of game we introduce here is designed to test whether agents with access to QC can be fundamentally superior to agents having only access to classical computation. We focus on fully observed zero-sum two-player games with terminal states. Familiar games in this category include Chess, Checkers, and Go. A terminal state S_T is a state where the game being played has ended and a result must be obtained to determine the outcome.

We examine the case where the environment in Fig. 1 contains a quantum computer, and the result of gameplay is $z = F(S_T)$ where F is any function where a claim of quantum advantage has been made. Here $z = \{+1, -1, 0\}$ is the game result, where $z = +1$ means player one won, $z = -1$ means player two won, and $z = 0$ is a draw. We will use the convention that player one is red, and player two is blue.

Tangled

To make this idea concrete, we introduce a game we call *Tangled*. Tangled is a fully observed zero-sum two-player game with terminal states.

Tangled is played on a connected game graph $\mathcal{G}(V, E)$, where V is a set of $|V|$ vertices and E is a set of $|E|$ edges $\{e_{ij}\}$ where e_{ij} is an edge connecting vertices i and j and $i < j$. Each player starts the game with one vertex owned by that player, which are V_k and V_m for players one and two respectively. The values of k and m are parameters of the game.

Players take turns coloring all the edges of $\mathcal{G}$ one of three colors: gray, green, or purple. Player one (red) always moves first. The game starts with all edges unselected. The game ends, and the winner is determined, once all edges have been colored. The final game graph state at the end of a game is the terminal state.

The intuition behind Tangled (which we will make precise in the following sections) is that vertices map to spins, and edge colors map to Ising couplings between spins, with green mapping to ferromagnetic interactions and purple mapping to antiferromagnetic interactions. Gray edges have zero coupling. The objective of the game is to have more of the spins in the graph align with the spin on your vertex than with the spin on the other player's vertex.

State Space

Each possible game state can be written as a list of $|E|$ integers each drawn from the set $\{0, 1, 2, 3\}$, corresponding respectively to uncolored, gray, green, and purple, giving a total of $4^{|E|}$ possible game states. There are a total of $3^{|E|}$ possible terminal states (zeroes (uncolored edges) not being allowed in terminal states).

The $|E|$ elements of the state list are assigned to the edges in lexical/dictionary order (for example, e_{01} is first in the list if it exists, then e_{02}, etc., all the way up to the last possible edge). The initial state of all games is a list of $|E|$ zeroes. The terminal state of each game has no 0 values in the state list (all entries one of 1, 2, or 3).

Action Space

There are $3|E|$ possible actions, corresponding to the 3 possible choices per edge.

Moves Per Game

Each Tangled game has exactly $|E|$ total moves.

Adjudication of Terminal States

Here we describe the function $z = F(S_T)$ that computes the winner of a Tangled game with terminal state S_T, which is a complete edge coloring of the graph $\mathcal{G}$.

We consider a time-dependent Hamiltonian that interpolates between a driving Hamiltonian $\mathcal{H}_D$ and a classical Ising problem Hamiltonian $\mathcal{H}_P$:

$$H(t) = \Gamma(t/t_a)\mathcal{H}_D + \mathcal{J}(t/t_a)\mathcal{H}_P \tag{1}$$

$$\mathcal{H}_D = -\sum_i \sigma_i^x, \quad \mathcal{H}_P = \sum_{e_{ij}\in E} J_{ij}\sigma_i^z\sigma_j^z, \tag{2}$$

where $\sigma_i^{x,z}$ are Pauli matrices acting on qubit i, and Γ and $\mathcal{J}$ are the transverse-field and Ising energy scales [20], respectively. The evolution starts from time $t = 0$ in a paramagnetic phase with $\Gamma(0) \gg \mathcal{J}(0)$ and ends at time $t = t_a$, which is a parameter of the adjudication, with $\Gamma(1) \ll \mathcal{J}(1)$, deep in the spin-glass phase. These two phases are separated by a quantum phase transition (QPT) whose behavior is dictated by the topology of the programmed Ising model.

Any game terminal state $S_T = (s_0, ..., s_{|E|-1})$ can be mapped onto an Ising model

$$E(s_0, ..., s_{|E|-1}) = \sum_{e_{ij}\in E} J_{ij}s_is_j \tag{3}$$

where each vertex V_i in the game graph is mapped to a spin s_i, and couplings between two vertices V_i and V_j are:

- $J_{ij} = 0$ if edge $e_{ij} \notin E$
- $J_{ij} = 0$ if $S[e_{ij}]$ = 1
- $J_{ij} = -1$ if $S[e_{ij}]$ = 2
- $J_{ij} = 1$ if $S[e_{ij}]$ = 3

where the notation $S[e_{ij}]$ means the value assigned to edge e_{ij} during gameplay (recall that the state definition has $(s_0, ..., s_{|E|-1})$ in dictionary order). Under this map, gray, green, and purple colorings correspond to zero, ferromagnetic, and antiferromagnetic couplings respectively.

We define the spin-spin correlation matrix after the above evolution to be

$$C_{ij} =< \sigma_i^z\sigma_j^z > - < \sigma_i^z >< \sigma_j^z > \tag{4}$$

from which we derive the expression for a quantity we call the *score* of the game

$$R = \sum_{j\neq k} C_{kj} - \sum_{j\neq m} C_{mj} \tag{5}$$

where k is the index of the vertex V_k owned by player 1, and m is the index of the vertex V_m owned by player 2.

We then define the evaluation of the terminal state to be

- $z = +1$ if $R > \epsilon$ (player 1 wins if $R > \epsilon$)
- $z = -1$ if $R < -\epsilon$ (player 2 wins if $R < -\epsilon$)
- $z = 0$ if $-\epsilon \leq R \leq \epsilon$ (draw if $|R| \leq \epsilon$)

where ϵ is a parameter of the adjudication.

The authors of [20] claim that computing the spin-spin correlation matrix C_{ij} at $s = 1$ after transition through the QPT is a domain of quantum supremacy for certain ranges of anneal time t_a. If this claim holds, then evaluation of Tangled terminal states also has this property.

Three Different Adjudication Strategies

Here we consider three different adjudication strategies.

1. The first is an exact solver that simulates the time evolution of Eq. 1 via integration of the Schrodinger Equation. This method is straightforward but limited to small game graphs due to the exponential cost of the method.

2. The second uses simulated annealing to approximate the outcome of the dynamical process, by attempting to return the equilibrium statistics of the classical Ising problem Hamiltonian. This approximation should work well in the regime where the system has time to thermally equilibrate into the ground state of the problem Hamiltonian post-anneal, but we expect it to fail when computation timescales are shorter than the timescale for thermal equilibration of the problem Hamiltonian, which in the spin glass phase should be the case (see [41] for analysis and experimental results supporting this). This second strategy is a specific example of a classical approximation to computing reward associated with terminal states $\bar{z} = \bar{F}(S_T)$.

3. The third uses the D-Wave Advantage2 quantum computing system, by embedding the game graph into hardware and collecting statistics on running the quantum annealing process described in Eq. 1 and thereby computing the score of the game (see [42] for an overview of the embedding process used). This third strategy is by definition the ground truth for adjudication of terminal states by a quantum computer $z = F(S_T)$.

Another possibility for adjudication are tensor networks [43], which model quantum systems by assuming entanglement between the system's degrees of freedom is local. The code described in this paper has not been released, and the methods described in the paper are non-trivial to reproduce independently. The trends shown in [20, 41, 43] indicate that the tensor network approach should fail in the quantum supremacy regime (the D-Wave system has strong evidence of volume-law entanglement entropy scaling [44]). This solver method should provide a better approximation to true reward for Tangled (and more generally, the domain of quantum advantage claimed by the quantum annealing approach), and will be tested if and when code is released.

Game Parameters

We denote a fully specified instance of Tangled $T(\mathcal{G}, (k, m), \epsilon, t_a)$, where the four parameters are:

1. The game graph $\mathcal{G}$
2. The choice of player-owned vertices V_k and V_m
3. The draw boundary ϵ
4. The anneal time t_a

Comparing Agent Capability

We use the Elo score [45], a measure of agent performance in standard use in the RL community (see also Appendix A).

Classical Simulation as a Corrupted Reward Signal

An aspect of RL that can be confusing is the nature of the boundary between agent and environment. When dealing with a classical agent and a classical environment, it is usually fine to not have to deal with where this boundary actually is, as the computation that's running the agent uses the same substrate as its environment (classical computing hardware), and the systems the agent models are also classical systems.

Information is always embodied, and there cannot be a concept of an agent outside of the matter and energy upon which its information resides. This means that there is no real boundary where 'agent' rests squarely on one side and 'environment' sits on the other. Decision-making and learning are physical processes that are inextricably intertwined with the physics of the substrate upon which they occur. In the purely classical case, we can safely ignore this as the laws governing the information that makes up the agent are the same as the laws of the environment it is learning about. But in our case, we have to be more careful about how we draw this boundary.

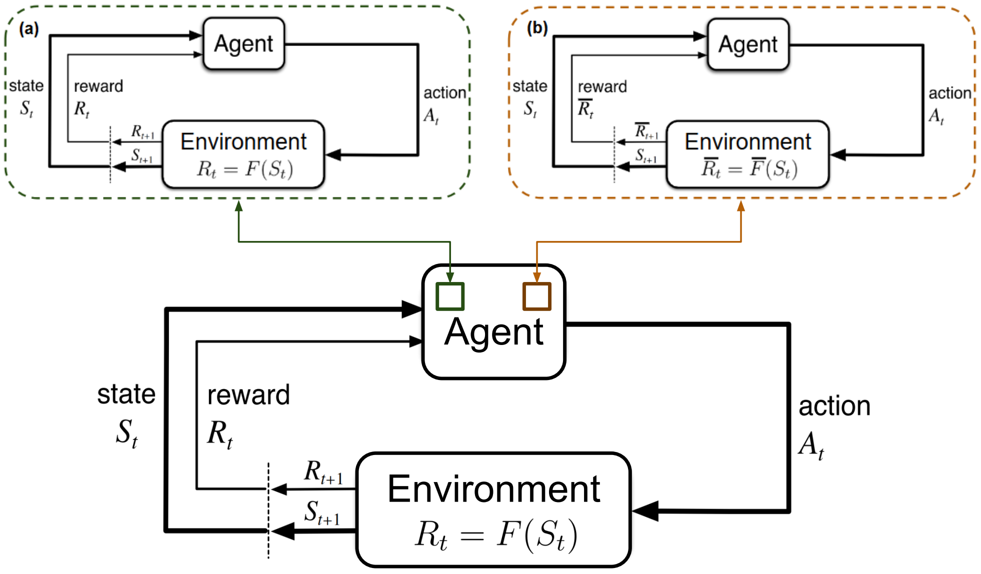


Figure 2: An agent in the real world (bottom) takes actions A_t that generate environment states S_t and true rewards $R_t = F(S_t)$. We assume that inside the agent there is a model of the world, where (a) the model includes QC, which is assumed to accurately compute $R_t = F(S_t)$, or (b) the model does not include QC, in which case the reward $\bar{R}_t = \bar{F}(S_t)$ is assumed to be corrupted, in the sense that $\bar{R}_t \neq R_t$ for at least some states S_t.

We are interested in cases where measuring $F(S_t)$ from the real world is expensive, such as measuring the properties of a novel small molecule drug. This makes techniques where an agent learns from directly sampling from the environment, such as inverse reinforcement learning [46], impractical, and an agent must attempt to learn policies from internal simulation. We differentiate the cases where the agent has access to the ground truth reward signal (Fig. 2a), or must rely on classical approximations to it (Fig. 2b). These two cases differ in that in (a) the agent resides at least in part on a quantum mechanical physical substrate, whereas in (b) the agent resides entirely on a substrate designed to be classical. We treat the latter case as learning in the presence of a corrupted reward signal [16].

In the case where the possible values of R_t are a discrete set (such as is the case for two-player zero-sum games with ($R_t \in \{+1, -1, 0\}$) or without ($R_t \in \{+1, -1\}$) draws), if there are no other possible simplifying assumptions on the corruption channels, the authors of [16] derive a No Free Lunch theorem proving that no possible agent learning from the corrupted reward signal can outperform a random agent. In this limit, one can prove that agents learning from the true reward signal by running at least in part on quantum mechanical substrates (agents of the type shown in Fig. 2a) are categorically superior to any possible classical agent.

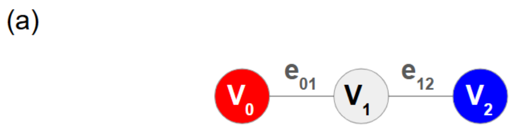
In the case studied here, where we use simulated annealing to compute $\bar{F}(S_t)$, the dominant error type is systematic mis-specification of reward. For example, if for some state S_t the true reward is $R_t = F(S_T) = +1$, simulated annealing might consistently produce $\bar{R}_t = \bar{F}(S_T) = -1$ (we will see in the following section that this is the case and provide a physical mechanism for it). This type of systematic reward corruption is especially dangerous, as the agent has no way of differentiating between cases where the classical approximation is reporting the true reward or reporting an incorrect reward.

Hello World: Tangled on P_3

To see how Tangled gameplay works, we start by playing on the smallest game graph that allows wins – P_3, the path graph on three vertices (Fig. 3a). Here we focus on the $T(P_3, (0, 2), 1/2, 350\text{ns})$ game instance. The full game tree is shown in Fig. 3b. Player one owns vertex V_0 (shown colored in red), and player two owns vertex V_2 (shown colored in blue). We use this coloring convention for vertex ownership throughout.

Player one moves first. They have $3|E| = 6$ possible actions, corresponding to coloring one of the edges e_{01} or e_{12} one of gray, green, or purple.

Once player one has chosen their action, player two must select the unselected edge, and color it one of gray, green, or purple. For this tiny two-edge graph, this creates a terminal state where all edges are colored, the game is over, and the terminal state must be adjudicated. There are $3^{|E|} = 3^2 = 9$ terminal states. We adjudicate each using the three different adjudicators described in the previous Section. Shown in Fig. 4 are the results for all nine terminal states for the three solvers for the choices of game parameters $t_a = 350\text{ns}$ and $\epsilon = 1/2$. We see that for this game graph and choice of parameters, none of the terminal states is near the $\epsilon = \pm 1/2$ draw boundaries, and all three solvers agree on the adjudication of all terminal states.



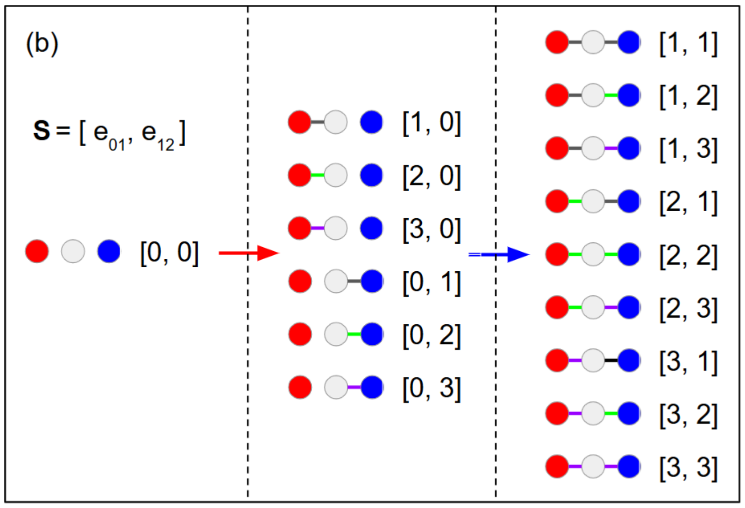


Figure 3: (a) The three-vertex two-edge ($|V| = 3$, $|E| = 2$) path graph P_3 is the smallest graph where one player can win. There are $4^{|E|} = 4^2 = 16$ possible game states, $3^{|E|} = 3^2 = 9$ terminal states, $3|E| = 6$ possible actions, and $|E| = 2$ total moves per game (one per player). (b) The entire game tree, showing game progression from initial state $S = [0, 0]$ on the left, to player one making one of six possible moves (middle), to player two making their move, ending up with nine possible terminal states on the right. Shown are both the graphs and their state representations S.

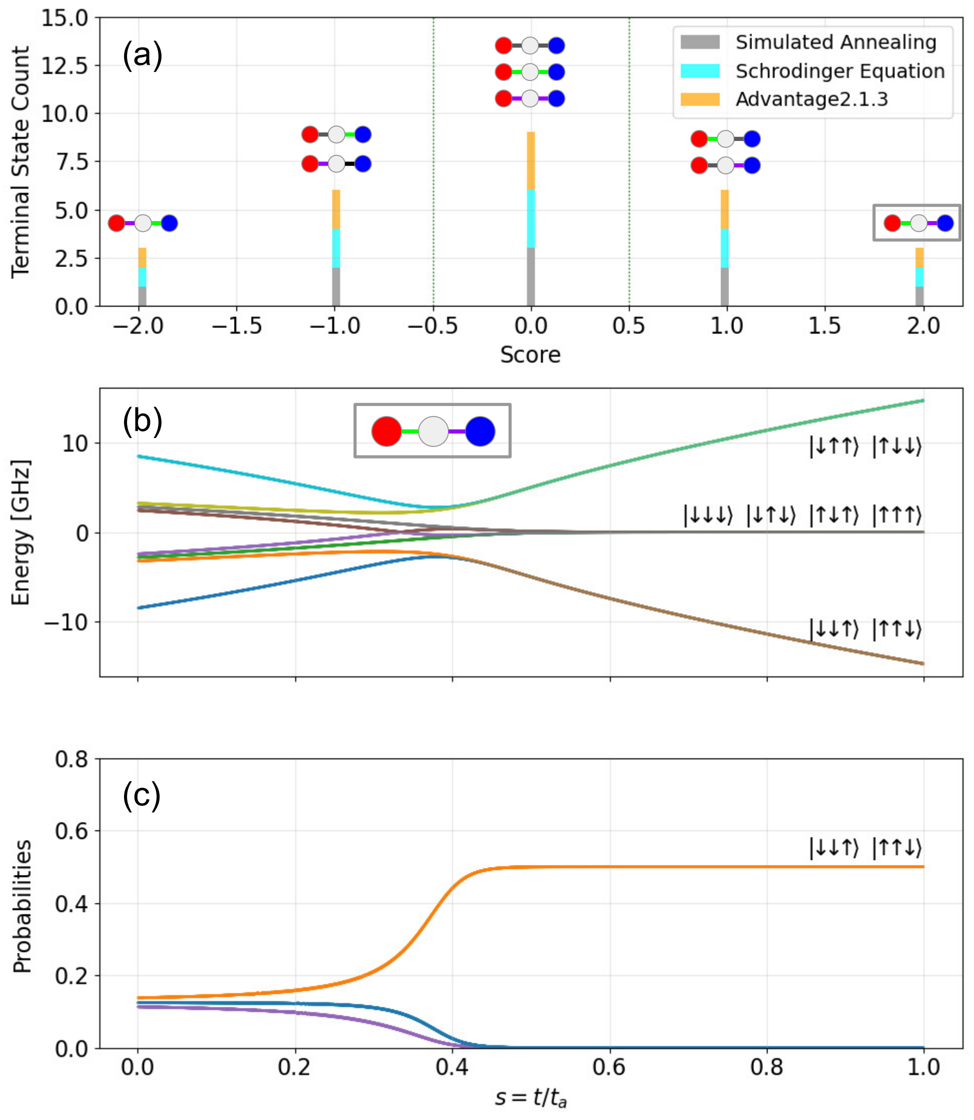


Figure 4: (a) Adjudication results for all nine terminal states with anneal time $t_a = 350\text{ns}$ and adjudication boundary $\epsilon = 1/2$ (shown as dashed green vertical lines). Scores less than $-\epsilon$ are wins for player two. Scores greater than ϵ are wins for player one. Scores with absolute value less than ϵ are draws. Histogram bins have width 0.01. (b) Energy eigenvalues for terminal state $S_T = [2, 3]$ (shown in inset) as a function of annealing parameter $s = t/t_a$ for $t_a = 350ns$, computed by solving the Schrodinger Equation using the annealing schedule of the D-Wave Advantage2.1.3 system [47]. At the beginning of the anneal, the system is in the unique ground state of $\mathcal{H}_D$. At the end of the anneal, the system Hamiltonian $\mathcal{H}_P$ is diagonal in the measurement basis. (c) Occupation probabilities for energy eigenstates as a function of annealing parameter s computed using the Schrodinger Equation. Near $s = 1$ the Hamiltonian is diagonal, and the occupation probabilities of the two ground states $|\downarrow\downarrow\uparrow\rangle$ and $|\uparrow\uparrow\downarrow\rangle$ are both 50%.

For the example terminal state $S_T = [2,3]$ shown in Fig. 4b, there are two equally likely degenerate ground states $|\downarrow\downarrow\uparrow\rangle$ and $|\uparrow\uparrow\downarrow\rangle$ (see Fig. 4c), which are assignments of spins to vertices that satisfy both edge constraints. We can then exactly calculate the correlation matrix using the Schrodinger Equation approach, and also approximate it by drawing samples from both the simulated and quantum annealers. All three approaches give

$$\hat{C} = \begin{bmatrix} 1.0 & 1.0 & -1.0 \\ 1.0 & 1.0 & -1.0 \\ -1.0 & -1.0 & 1.0 \end{bmatrix} \tag{6}$$

with the resultant score

$$R = \sum_{j\neq 0} C_{0j} - \sum_{j\neq 2} C_{2j} = (C_{01} + C_{02}) - (C_{20} + C_{21}) = +2.0 \tag{7}$$

Since the score is positive, player one wins. To understand intuitively why player one wins, note that examination of the two ground states shows that in both cases the middle vertex's spin aligns with player one's vertex's spin, whereas in the case of player two the middle spin is anti-aligned. Therefore for all samples drawn at $s = 1$, the sum over the correlations of player one's spin will be larger than player two's.

We can see by inspection of Fig. 4a that if player one sets edge e_{01} to green or edge e_{12} to purple, they are guaranteed at worst a draw. Given either of these first moves, player two forces a draw by setting the remaining edge to the same color as player one chose. This is the optimal strategy for both players for this game instance.

Introducing Frustration: Tangled on K_3

Here we examine the Tangled instance $T(K_3, (0,2), 1/2, 350\text{ns})$. The fully connected graph on three vertices K_3 introduces the possibility of *frustration* into Tangled. A frustrated system is one in which not all edge constraints can be met by any assignment of spin directions to the vertices.

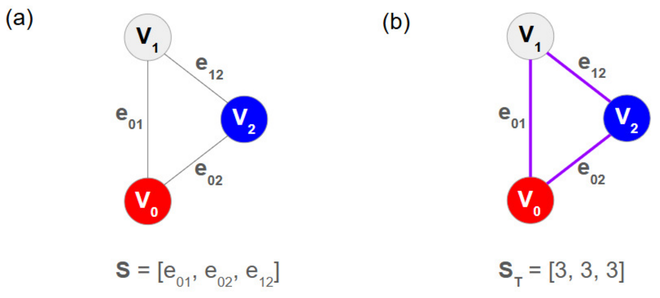
Shown in Fig. 5b is an example of a terminal state where the system is frustrated. Recall that purple links map to antiferromagnetic couplings between spin variables on the vertices. For this specific terminal state, Eq. (1) is

$$H(t) = \Gamma(t/t_a)\mathcal{H}_D + \mathcal{J}(t/t_a)\mathcal{H}_P \tag{8}$$

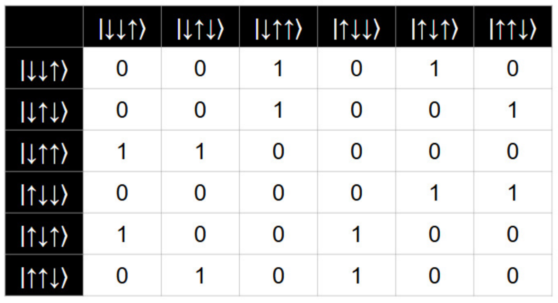
$$\mathcal{H}_D = -\sigma_0^x - \sigma_1^x - \sigma_2^x, \quad \mathcal{H}_P = +\sigma_0^z\sigma_1^z + \sigma_0^z\sigma_2^z + \sigma_1^z\sigma_2^z \tag{9}$$

Frustrated quantum magnets can exhibit an effect called *order-by-disorder* [48, 49, 50, 51, 52, 53, 54]. This effect arises from terms in the Hamiltonian that allow tunneling between states (such as the transverse term in Eq. (1)) lifting classical degeneracies and increasing the probability of being in specific classically degenerate states over others (which is the origin of the 'order' in order-by-disorder). This effect will be very important in gaining an intuition for why the adjudication process in Tangled is difficult to classically simulate (for an introduction to order-by-disorder relevant for Tangled, see [41]).

The terminal state in Fig. 5b has a ground state manifold containing six classically degenerate ground states. Shown in Fig. 5c is the adjacency matrix for these six states, where the entries are 1 if the states are separated by one spin flip and 0 otherwise.



(c)



	$\vert\downarrow\downarrow\uparrow\rangle$	$\vert\downarrow\uparrow\downarrow\rangle$	$\vert\downarrow\uparrow\uparrow\rangle$	$\vert\uparrow\downarrow\downarrow\rangle$	$\vert\uparrow\downarrow\uparrow\rangle$	$\vert\uparrow\uparrow\downarrow\rangle$
$\vert\downarrow\downarrow\uparrow\rangle$	0	0	1	0	1	0
$\vert\downarrow\uparrow\downarrow\rangle$	0	0	1	0	0	1
$\vert\downarrow\uparrow\uparrow\rangle$	1	1	0	0	0	0
$\vert\uparrow\downarrow\downarrow\rangle$	0	0	0	0	1	1
$\vert\uparrow\downarrow\uparrow\rangle$	1	0	0	1	0	0
$\vert\uparrow\uparrow\downarrow\rangle$	0	1	0	1	0	0

Figure 5: (a) The 3-vertex 3-edge complete graph on three vertices K_3, with state explicitly shown as functions of edge values in lexical order. (b) The terminal state $S_T = [3, 3, 3]$. (c) The adjacency matrix for the six classically degenerate ground states for this terminal state.

All of the ground states are one spin flip away from two others. In this case, there is no symmetry breaking and therefore no order-by-disorder, and all ground states should be equally populated. Unbiased sampling using simulated annealing should give the same result as the two quantum approaches. This is what we observe (shown in Fig. 6 are the results of adjudication on all terminal states, showing full agreement between the three approaches).

We can, like in the case of P_3, enumerate all terminal states and find optimal gameplay strategies by observation (see [55]). The optimal play is for red to first color e_{12} purple or e_{01} green, which guarantees red at worst a draw. Blue then forces a draw by either playing e_{12} green or e_{01} purple, making red's last move not change the outcome.

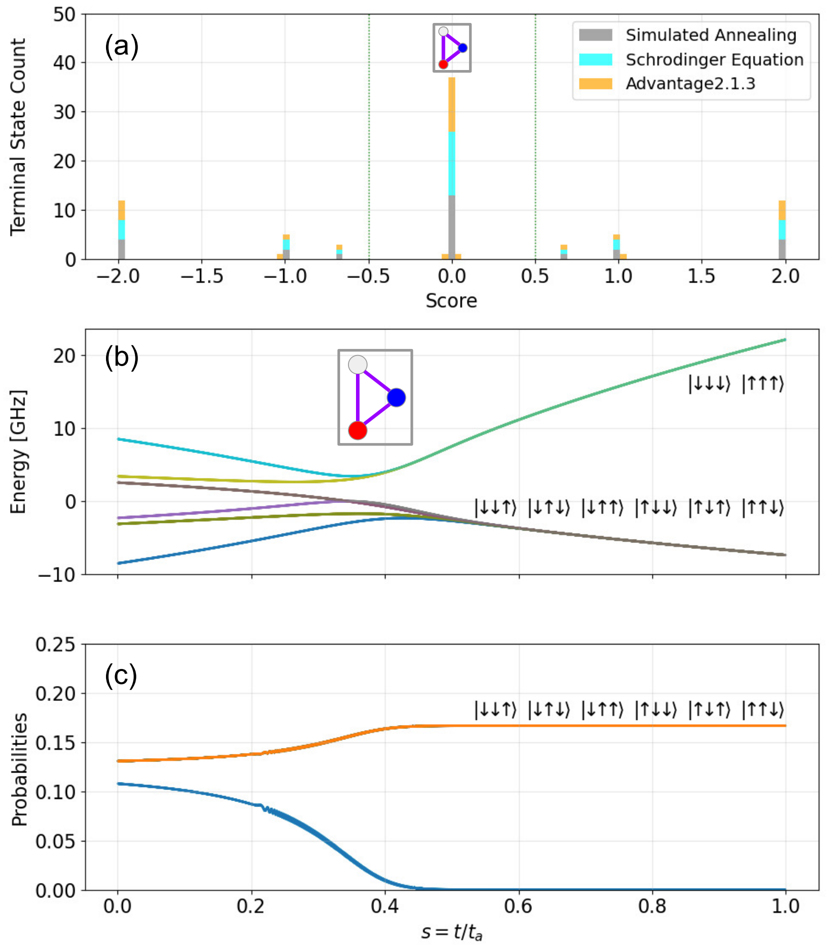


Figure 6: (a) Adjudication results for all $3^{|E|} = 3^3 = 27$ terminal states of K_3 for all three solvers, with anneal time $t_a = 350\text{ns}$ and draw boundary $\epsilon = 1/2$, with the state $S_T = [3, 3, 3]$ with score ~ 0 explicitly shown. (b) Energy eigenvalues (obtained from the Schrodinger Equation) for the terminal state $S_T = [3, 3, 3]$ (shown in inset) as a function of annealing parameter $s = t/t_a$. (c) Occupation probabilities for $S_T = [3, 3, 3]$ from the Schrodinger Equation as a function of annealing parameter $s = t/t_a$. All six classically degenerate ground states have the same (1/6) probability in the $s = 1$ limit, as in this case all six states are symmetric (see Fig. 5c).

Order-by-Disorder: Tangled on the Diamond Graph

While K_3 has frustrated terminal states, this didn't lead to any interesting effects. Here we look at a slightly bigger graph, the diamond graph $K_4 - e$, which is the fully connected graph on four vertices with one edge removed (see Fig. 7a). This graph has $3^{|E|} = 3^5 = 127$ terminal states. Specifically we examine the $T(K_4 - e, (0, 2), 1/2, 350\text{ns})$ instance.

This game instance has something new – terminal states that adjudicate to different values when using simulated annealing versus the two quantum approaches. As an example of this, consider the terminal state $S_T = [1, 2, 2, 2, 3]$ shown in Fig. 7b. For this terminal state, Eq. (1) is

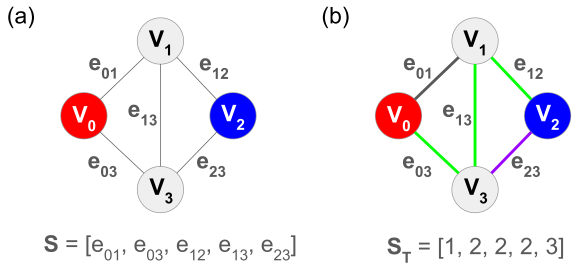
$$H(t) = \Gamma(t/t_a)\mathcal{H}_D + \mathcal{J}(t/t_a)\mathcal{H}_P \tag{10}$$

$$\mathcal{H}_D = -\sigma_0^x - \sigma_1^x - \sigma_2^x - \sigma_3^x, \quad \mathcal{H}_P = -\sigma_0^z\sigma_3^z - \sigma_1^z\sigma_2^z - \sigma_1^z\sigma_3^z + \sigma_2^z\sigma_3^z \tag{11}$$

giving a ground state manifold of six states ($|\downarrow\downarrow\downarrow\downarrow\rangle$, $|\downarrow\downarrow\uparrow\downarrow\rangle$, $|\downarrow\uparrow\uparrow\downarrow\rangle$, $|\uparrow\downarrow\downarrow\uparrow\rangle$, $|\uparrow\uparrow\downarrow\uparrow\rangle$, and $|\uparrow\uparrow\uparrow\uparrow\rangle$, see Fig. 8c).

If all six ground states are equally likely, we can exactly calculate the score for this case. The result is $+4/3$, which is what we obtain using the simulated annealing adjudicator (see Fig. 8b).

This is not the score we get for this state using the two quantum approaches – we instead get scores of $+2$. The reason for this is that this instance exhibits order-by-disorder, which changes the relative occupation probabilities of the classically degenerate ground states (see Fig. 7c and



(c)

	$|\downarrow\downarrow\downarrow\downarrow\rangle$	$|\downarrow\downarrow\uparrow\downarrow\rangle$	$|\downarrow\uparrow\uparrow\downarrow\rangle$	$|\uparrow\downarrow\downarrow\uparrow\rangle$	$|\uparrow\uparrow\downarrow\uparrow\rangle$	$|\uparrow\uparrow\uparrow\uparrow\rangle$
$|\downarrow\downarrow\downarrow\downarrow\rangle$	0	1	0	0	0	0
$|\downarrow\downarrow\uparrow\downarrow\rangle$	1	0	1	0	0	0
$|\downarrow\uparrow\uparrow\downarrow\rangle$	0	1	0	0	0	0
$|\uparrow\downarrow\downarrow\uparrow\rangle$	0	0	0	0	1	0
$|\uparrow\uparrow\downarrow\uparrow\rangle$	0	0	0	1	0	1
$|\uparrow\uparrow\uparrow\uparrow\rangle$	0	0	0	0	1	0

Figure 7: (a) The 4-vertex 5-edge diamond graph, with state explicitly shown as functions of edge values in lexical order. (b) The terminal state $S_T = [1, 2, 2, 2, 3]$. (c) The adjacency matrix for the six degenerate ground states for this terminal state. Unlike K_3, we see here that $|\downarrow\downarrow\uparrow\downarrow\rangle$ and $|\uparrow\uparrow\downarrow\uparrow\rangle$ are connected by a single spin flip to two other states, whereas the others are only connected to one other state.

Fig. 8c). With these new probabilities we can again calculate the score analytically and we obtain $+2$. Note that the quantum annealer reproduces this result (Fig. 8b).

In this case, and in all of the cases for the diamond graph with choices of game parameters used here where this effect occurs, the difference in scores between adjudicators doesn't change the outcome of the game. Even though we are seeing different scores, all three adjudicators agree on all 127 terminal states. While not as obvious as the previous P_3 and K_3 graphs, it is straightforward to find optimal play strategies for this game instance that force draws.

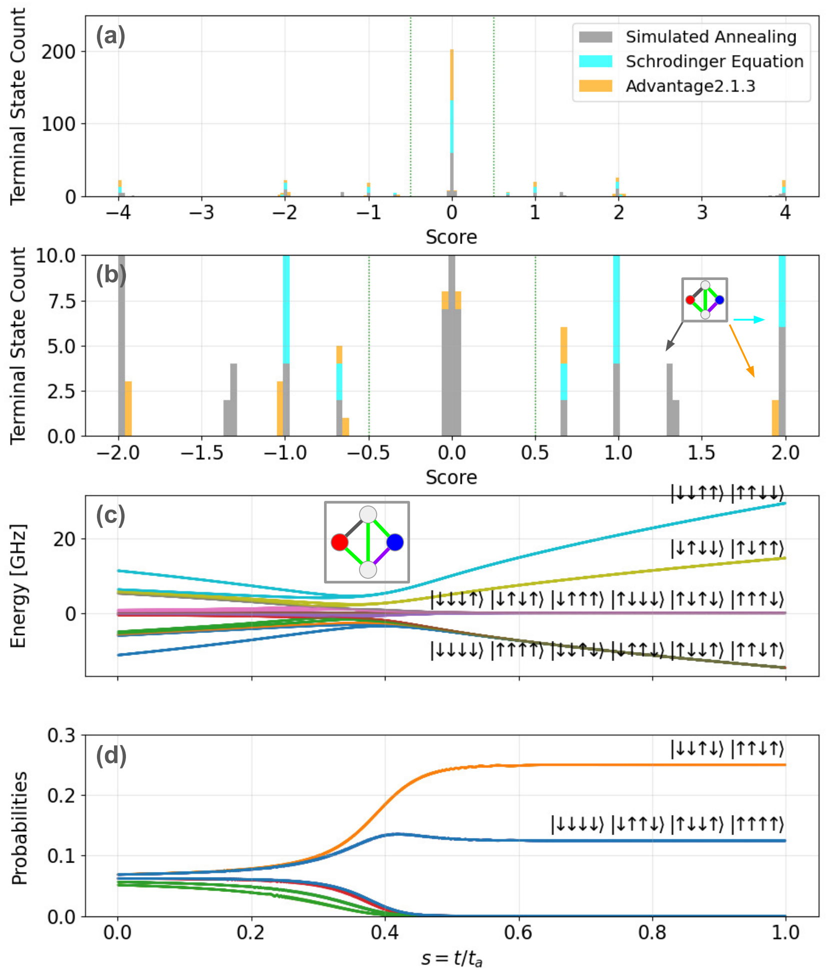


Figure 8: (a) Adjudication results for all $3^{|E|} = 3^5 = 127$ terminal states of the diamond graph for all three solvers, with anneal time $t_a = 350\text{ns}$ and adjudication boundary $\epsilon = 1/2$. (b) Same data, but with change of axes scales, showing $S_T = [1, 2, 2, 2, 3]$ in the inset. This state adjudicates to $+4/3$ using simulated annealing, and $+2$ using both Schrodinger Equation and quantum annealing approaches (all wins for red). (c) Energy eigenvalues from the Schrodinger Equation for the terminal state $S_T = [1, 2, 2, 2, 3]$ (shown in inset) as a function of annealing parameter $s = t/t_a$. (d) Occupation probabilities for the same state obtained from the Schrodinger Equation, showing preferential occupation of $|\downarrow\downarrow\uparrow\downarrow\rangle$ and $|\uparrow\uparrow\downarrow\uparrow\rangle$ (each $p = 1/4$) versus the other classically degenerate ground states (each $p = 1/8$).

Adjudication Errors: Tangled on The Barbell Graph

Here we examine gameplay on the barbell graph B_3 (Fig. 9a), which has $|V| = 6$ vertices and $|E| = 7$ edges, using the Tangled instance $T(B_3, (0, 4), 1/4, 350\text{ns})$. There are $4^{|E|} = 4^7 = 16,384$ game states and $3^{|E|} = 3^7 = 2,187$ terminal states, one of which is shown in Fig. 9b. There are $3|E| = 21$ possible actions, and $|E| = 7$ total moves per game (four for red, three for blue). The number of terminal states is small enough to adjudicate via enumeration for all three of our adjudication strategies. The results of this are shown in Fig. 10a.

While the Schrodinger Equation and quantum annealing adjudication results agree for all terminal states, the simulated annealing solver disagrees with both for 36 out of the $2,187$ ($\sim 1.7\%$) possible terminal states. This reward corruption is deterministic and its details are shown in Table 1. We examined these states and found that all of them contained frustration of one or both of the triangles in the barbell graph, and that the order-by-disorder effect we saw in the last Section quantitatively explained the disagreement in every case. In the example terminal state in Fig. 9b, it is easy to verify that both triangles are frustrated. This frustration leads, in this case, to an 18-fold classical ground state degeneracy (3 from the left triangle times 3 from the right triangle times two because of the $\uparrow\rightleftharpoons\downarrow$ symmetry of the Ising Hamiltonian).

This is one of the states where simulated annealing returns an incorrect adjudication. Assuming unbiased sampling, the score can be analytically calculated to be $-4/9$, which is the value returned by simulated annealing (see Fig. 10b). We can see via solving the Schrodinger Equation that the probabilities of the 18 ground states are altered for the quantum cases (Fig. 10d), returning a score of $+1/2$. This is also the value returned by the quantum annealer.

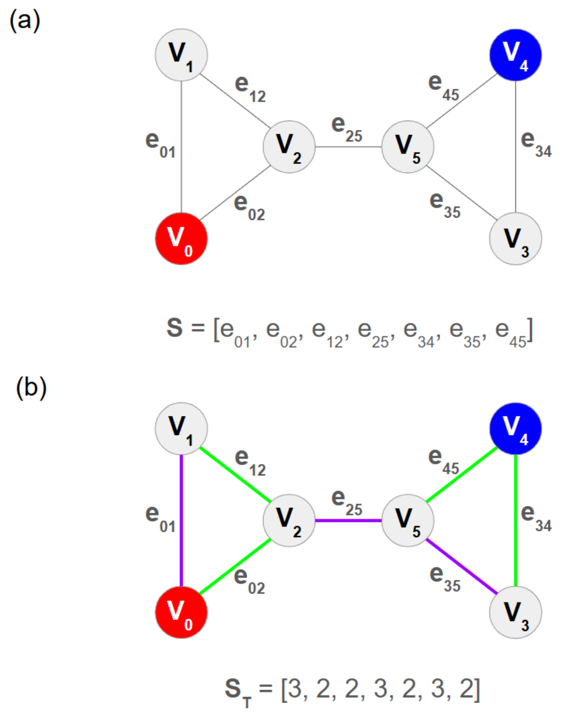


Figure 9: (a) The 6-vertex 7-edge barbell graph, with state explicitly shown as functions of edge values in lexical order. (b) The terminal state $S_T = [3, 2, 2, 3, 2, 3, 2]$.

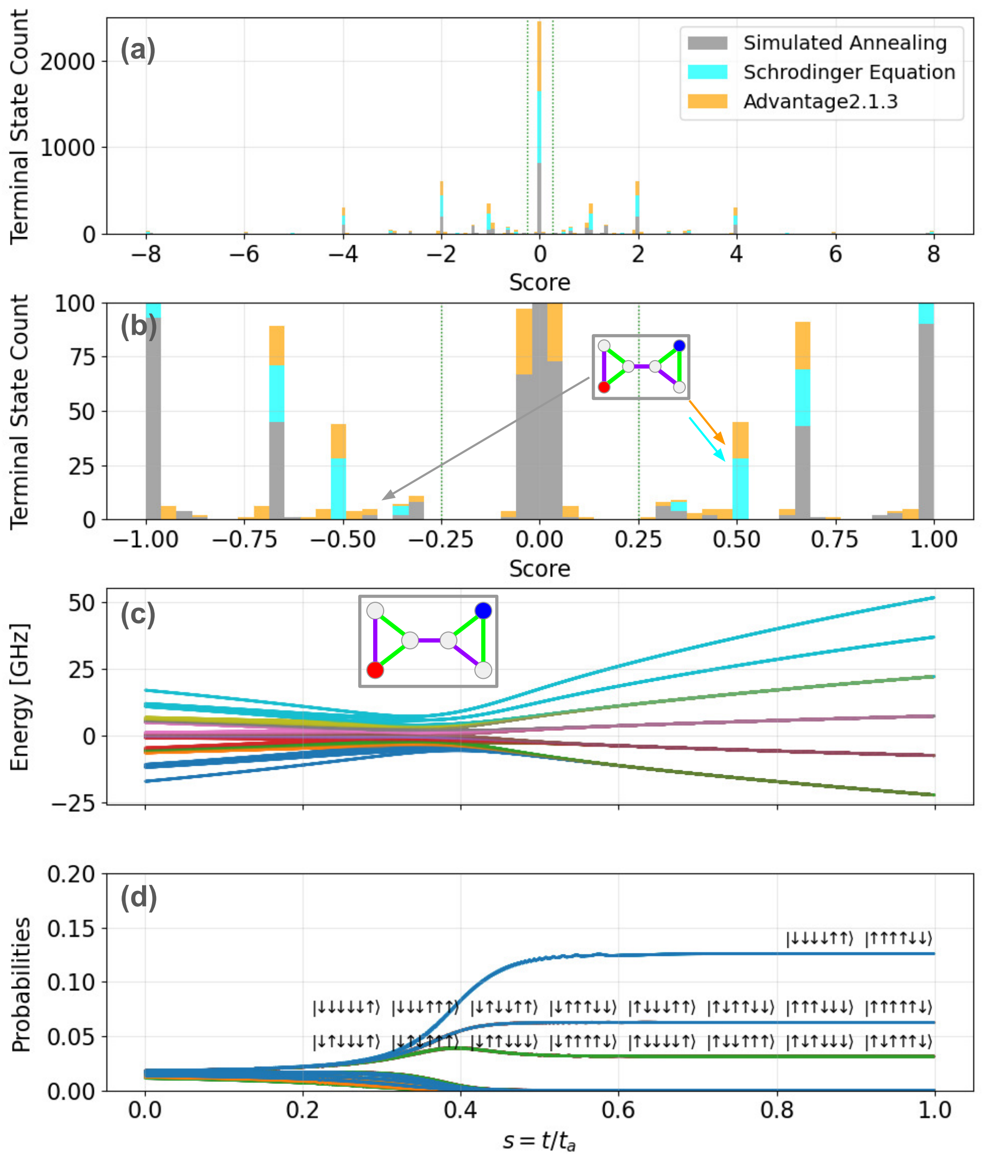


Figure 10: (a) Adjudication results for all $3^{|E|} = 3^7 = 2,187$ terminal states of the barbell graph for all three solvers, with anneal time $t_a = 350\text{ns}$ and adjudication boundary $\epsilon = 1/4$. Simulated annealing generated 36 states ($\sim 1.7\%$) with incorrect adjudication, whereas the Schrodinger Equation solver matched the quantum annealing adjudication for all terminal states. (b) Same data, but with change of axes scales, showing $S_T = [3, 2, 2, 3, 2, 3, 2]$ in the inset. This state adjudicates to $-4/9$ using simulated annealing (win for blue), and $+1/2$ using both Schrodinger Equation and quantum annealing approaches (win for red). (c) Energy eigenvalues from the Schrodinger Equation for the terminal state $S_T = [3, 2, 2, 3, 2, 3, 2]$ (shown in inset) as a function of annealing parameter $s = t/t_a$. (d) Occupation probabilities for the same state obtained from the Schrodinger Equation, showing splitting of occupation probabilities among the ground state manifold.

Unlike for the diamond graph, here order-by-disorder not only changes the value of the score, but it also changes the result of the adjudication for some of the terminal states. The simulated annealer scores this particular terminal state for player two ($\bar{R} = \bar{F}(S_T) = -1$), which is incorrect. The correct adjudication for this state is a win for player one ($R = F(S_T) = +1$; recall the ground truth is provided by the quantum annealer).

Here we find our first instance of reward corruption and have quantified its cause. Specifically, if we use simulated annealing to approximate $R = F(S_T)$ for Tangled, we find 36 out of 2,187 terminal states where $\bar{R} \neq R$, where the error mis-specification is deterministic and caused by the classical algorithm we used not taking into account order-by-disorder effects.

		Simulated Annealing		
		Draw	Red	Blue
Quantum Annealing	Draw	—	6	10
	Red	10	—	2
	Blue	6	2	—

Table 1: Reward corruption for the Barbell graph. There are four states where simulated annealing reverses the sign on the reward $+1 \leftrightarrow -1$, and 32 where reward is mislabeled $0 \leftrightarrow \pm 1$.

Increasing Adjudication Error Fraction: Tangled on The 3-Prism

For the Barbell graph, simulated annealing fails for a small fraction of possible terminal states. Here we investigate a slightly bigger graph with a larger fraction of states where simulated annealing gives incorrect answers.

The 3-Prism Y_3 (Fig. 11a) has $|V| = 6$ vertices and $|E| = 9$ edges. We investigate the Tangled instance $T(Y_3, (0,4), 0.125, 100\text{ns})$. There are $4^{|E|} = 4^9 = 262{,}144$ game states and $3^{|E|} = 3^9 = 19{,}683$ terminal states, one of which is shown in Fig. 11b. There are $3|E| = 27$ possible actions, and $|E| = 9$ total moves per game (five for red, four for blue). We enumerated reward for all terminal states using both hardware quantum annealing and simulated annealing. Simulated annealing incorrectly adjudicated $2{,}175$ out of $19{,}683$ ($\sim 11.1\%$) terminal states (see Table 2).

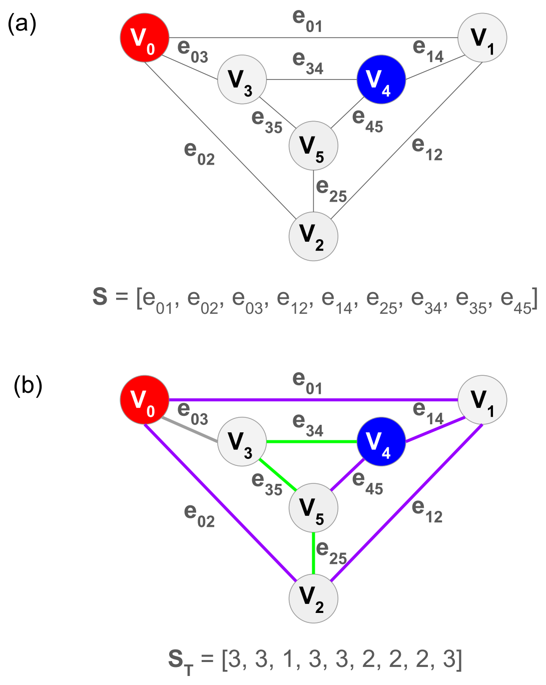

(a)
V0
V1
V2
V3
V4
V5
e01
e02
e03
e12
e14
e25
e34
e35
e45
S = [e01, e02, e03, e12, e14, e25, e34, e35, e45]
(b)
ST = [3, 3, 1, 3, 3, 2, 2, 2, 3]

Figure 11: (a) The 6-vertex 9-edge 3-Prism, with state explicitly shown as functions of edge values in lexical order. (b) The terminal state $S_T = [3, 3, 1, 3, 3, 2, 2, 2, 3]$.

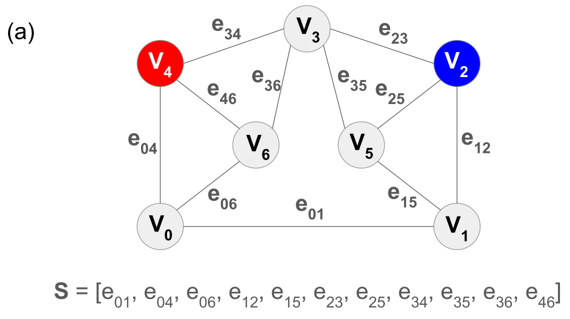
		Simulated Annealing		
		Draw	Red	Blue
Quantum Annealing	Draw	—	931	116
	Red	115	—	39
	Blue	938	36	—

Table 2: Reward corruption for the 3-Prism graph. There are 75 states where simulated annealing reverses the sign on the reward $+1 \leftrightarrow -1$, and $2{,}100$ where reward is mislabeled $0 \leftrightarrow \pm 1$.

Even More Errors: Tangled on The Moser Spindle

For the Barbell and 3-Prism graphs, the simulated annealing approach to adjudication fails for $\sim 1.7\%$ and $\sim 11.1\%$ of possible terminal states respectively. Here we investigate a bigger graph with a larger fraction of states where simulated annealing gives incorrect answers.

The Moser Spindle M_7 (Fig. 12a) has $|V| = 7$ vertices and $|E| = 11$ edges. This graph contains 3-, 4-, and 5-cycles that can exhibit frustration depending on the bond configuration. We investigate the Tangled instance $T(M_7, (2, 4), 0.05, 40\text{ns})$. There are $4^{|E|} = 4^{11} = 4, 194, 304$ game states and $3^{|E|} = 3^{11} = 177, 147$ terminal states, one of which is shown in Fig. 12b. There are $3|E| = 33$ possible actions, and $|E| = 11$ total moves per game (six for red, five for blue). We adjudicated all terminal states with both hardware quantum annealing and simulated annealing. Simulated annealing incorrectly adjudicated $43, 965$ out of the $177, 147$ ($\sim 24.8\%$) terminal states.



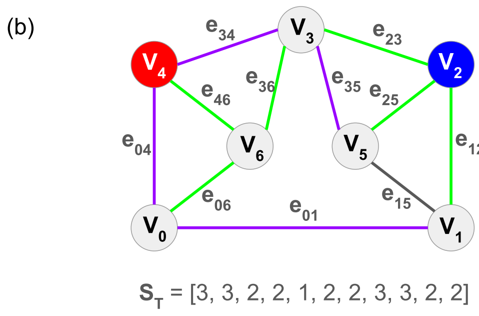


Figure 12: (a) The 7-vertex 11-edge Moser Spindle, with state explicitly shown as functions of edge values in lexical order. (b) The terminal state $S_T = [3, 3, 2, 2, 1, 2, 2, 3, 3, 2, 2]$.

		Simulated Annealing		
		Draw	Red	Blue
Quantum Annealing	Draw	—	17,061	2,743
	Red	2,636	—	2,200
	Blue	17,083	2,242	—

Table 3: Reward corruption for the Moser Spindle. There are $4, 442$ states where simulated annealing reverses the sign on the reward $+1 \leftrightarrow -1$, and $39, 523$ where reward is mislabeled $0 \leftrightarrow \pm 1$.

We now have three game instances with systematic reward corruption, one where the fraction is very small (the Barbell graph), one where the fraction is larger (the 3-Prism), and one where the fraction is even larger (the Moser Spindle graph). In the next Section, we introduce game-playing agents for Tangled, how we compare their relative capabilities, and quantify the effect of systematic adjudication errors on the performance of agents trained to play this game.

Agents

There are many possibilities for building game-playing agents for this type of game. There are three that we will focus on here. All of these are of the sort described in Fig. 1.

Random Agents

The first are agents that make random action choices at every move. Let $A(\vec{S})$ denote the set of available actions a in state $\vec{S}$. The random policy π_{random} is defined by:

$$\pi_{\text{random}}(a|\vec{S}) = \begin{cases} \frac{1}{|A(\vec{S})|} & \text{if } a \in A(\vec{S}) \\ 0 & \text{if } a \notin A(\vec{S}) \end{cases} \tag{12}$$

Monte Carlo Tree Search Agents

The second are agents that use online Monte Carlo Tree Search (MCTS) [56, 57] to select actions. MCTS is a heuristic search algorithm that builds a search tree incrementally and asymmetrically, focusing computational resources on the most promising paths.

We use the Upper Confidence Trees (UCT) policy [37] during the selection phase with exploration parameter $c = 2$ and the random policy Eq. 12 during the simulation phase. MCTS is parametrized by the number of rollouts M per action selection, and requires one terminal state adjudication per rollout. Because of this, our MCTS agents are also parametrized by the choice of adjudicator and its parameters.

We write MCTS[M, H] to denote an online MCTS agent using M rollouts with terminal state adjudicator $H \in \{SE, SA, QA\}$, where SE is the Schrodinger Equation, SA is simulated annealing, and QA is quantum annealing using the D-Wave Advantage2 system (parameters of each of these solvers are suppressed in this notation, but will be made explicit when we use this policy in what follows).

The MCTS policy is

$$\pi_{MCTS}(a|\vec{S};\tau) = \frac{N(\vec{S},a)^{(1/\tau)}}{\sum_b N(\vec{S},b)^{(1/\tau)}} \tag{13}$$

where $\pi_{MCTS}(a|\vec{S};\tau)$ is the probability of selecting action a from state $\vec{S}$, $N(\vec{S},a)$ is the number of times action a was visited during MCTS rollouts from state $\vec{S}$, and τ is a temperature parameter controlling how deterministic the algorithm is. In our implementation we chose $\tau = 0$ for MCTS agents, which reduces the MCTS policy to

$$\pi_{MCTS}(a|\vec{S},\tau = 0) = \arg\max_a N(\vec{S},a) \tag{14}$$

AlphaZero Agents

The third are AlphaZero agents [58]. For fully observed zero-sum two-player games with terminal states, AlphaZero is a powerful and simple (although computationally expensive) algorithm for producing high-quality agents.

AlphaZero employs a self-play reinforcement learning framework that combines MCTS with a neural net $f_{\vec{\theta}}(\vec{S}) = (\vec{p}, v)$, where $\vec{\theta}$ are the neural net's parameters, and $\vec{p}$ and v are action probabilities and value respectively for a given state $\vec{S}$. The algorithm iteratively improves through self-play, where moves are selected via MCTS guided by the Polynomial Upper Confidence Tree (PUCT) policy [59, 60] with exploration parameter $c = 2$. During self-play, the algorithm performs M_0 MCTS rollouts per move and selects moves according to the MCTS policy Eq. 13. During the competitive phase, the number of MCTS rollouts can be different and is denoted M_1. The temperature parameter τ decreases from initial exploration ($\tau = 1$) for the first P moves to deterministic play ($\tau = 0$) for the last $|E| - P$ moves during endgame, where P is a training parameter. We fix $P = |E| - 2$, so that only the last two moves use $\tau = 0$. As in pure MCTS, the terminal state adjudicator H is a parameter of the training process.

Training data consists of (state $\vec{S}$, MCTS policy π_{MCTS}, game outcome z) tuples collected from self-play, which are used to optimize the neural network parameters $\vec{\theta}$ via gradient descent with typical learning rates of 10^{-3} to 10^{-4}. For Tangled, $z \in \{0, +1, -1\}$, and for sufficiently large game graphs will be difficult to compute if not using D-Wave hardware if the quantum supremacy claim is valid. The training data, when $H \neq QA$, should contain adjudication errors that will corrupt the training process.

The neural network architecture we use here is a dual-head (policy and value outputs) deep residual convolutional architecture inspired by ResNet [61], consisting of multiple residual blocks with skip connections. ResNet architectures were originally designed for computer vision tasks. They are effective in the AlphaZero context, likely because most board game states are designed to be processed visually by people, but games like Tangled might benefit from architectures designed for graph inputs. It is possible that other architectures could be interesting to try in the Tangled context but we do not pursue this here.

We distinguish two different phases for AlphaZero agents. The first phase is the training phase, where the system's neural net is trained. Training requires setting the following parameters: the neural net, parametrized by $\vec{\theta}$; the number of MCTS rollouts per action selection during self-play and competitive phases M_0 and M_1 respectively; the terminal state adjudicator H and its parameters; the number of iterations performed N_{it}; the number of complete self-play games to simulate during one iteration N_{eps}; an Elo update threshold U for accepting a new neural net after competitive play; and the number of games to play during the competitive phase N_g. Here we fix $M_0 = 100$, $M_1 = 10$, $N_{it} = 50$, $N_{eps} = 3,200$, $U = 1$, and $N_g = 2,368$. Implementation details and parameters used are available upon request.

Once the training process is complete and we are ready to use an AlphaZero agent, we need to specify the number of MCTS rollouts to use per action selection. In the case of AlphaZero, the rollout action selection probabilities are guided by the trained neural net. We denote a trained AlphaZero agent AZ[M, H] where M is the number of MCTS rollouts performed during use of the agent, and H is the adjudicator used to train the agent.

Agent Performance

Barbell Graph

The Tangled instance $T(B_3, (0, 4), 1/4, 350\text{ns})$, where B_3 is the Barbell Graph, contains 36 out of $2, 187$ ($\sim$ 1.7%) terminal states where simulated annealing corrupts reward (see Table 1). Intuitively, this should make agents trained using the simulated annealing adjudicator perform slightly worse than those trained using quantum annealing.

We can estimate the difference in expected win rates between agents trained with or without corrupted reward, under the assumption that all terminal states are equally likely to be seen as endgame states. Define effective win rate using the true reward as $E_w = p_w + 0.5p_d$ where p_w and p_d are win and draw probabilities respectively, and effective win rate in the presence of reward corruption as $E^c_w = p^c_w + 0.5p^c_d$ where p^c_w and p^c_d are win and draw probabilities achieved using agents trained on the corrupted reward. Then the difference in effective win rates $E_w - E^c_w = p_w - p^c_w + 0.5(p_d - p^c_d) \approx p_{+1\Leftrightarrow-1} + 0.5 * p_{0\Leftrightarrow\pm1}$. For the Barbell graph, this is $E_w - E^c_w \approx 4/2187 + 0.5 * 32/2187 \approx 0.9\%$. Note that depending on the structure of both the agents and the game, this estimate may not hold. As an example, an agent that plays preferentially to end up in some small subset of terminal states will not see the statistics of all the terminal states and could perform much better or much worse than this.

Here we empirically investigate differences in performance of a set of agents by conducting a round-robin tournament. We distinguish between the adjudicator used to generate actions (the H parameters in the agents) and the adjudicator used to adjudicate the result of gameplay (which is fixed to be hardware-based quantum annealing, as by definition this is the ground truth for adjudication of Tangled). This implements the concept in Fig. 2, where ground truth evaluation of reward occurs from the environment (in our case, via hardware-based quantum annealing), whereas both training the agents (in the case of AlphaZero) and 'thinking about what moves to make' (online MCTS rollouts during game play for both MCTS and AlphaZero) is done internally inside the agents, where the quantum annealing-based agents are of type (a) and the simulated annealing-based agents are of type (b).

In each agent vs. agent competition, we played 4,992 games with the first agent as red, and

Player	**MCTS[100,SA]**	**Random**	**MCTS[10,SA]**	**MCTS[10,QA]**	**MCTS[100,QA]**	**AZ[10,SA]**	**AZ[10,QA]**	**AZ[100,SA]**	**AZ[100,QA]**	**Elo**
Random	117-9004-863	—	632-6692-2660	549-6910-2525	12-9059-913	28-9413-543	0-9393-591	46-9486-452	0-9493-491	515.2 ± 1.2
MCTS[10,SA]	559-5674-3751	6692-632-2660	—	2668-3145-4171	236-5788-3960	139-7150-2695	0-6976-3008	174-7299-2511	0-7453-2531	788.8 ± 1.2
MCTS[10,QA]	1116-5270-3598	6910-549-2525	3145-2668-4171	—	220-5804-3960	292-7211-2481	0-7065-2919	366-7238-2380	0-7524-2460	802.4 ± 1.2
MCTS[100,SA]	—	9004-117-863	5674-559-3751	5270-1116-3598	1421-1641-6922	5-2776-7203	0-2598-7386	11-2907-7066	0-2625-7359	1000.0
MCTS[100,QA]	1641-1421-6922	9059-12-913	5788-236-3960	5804-220-3960	—	235-2737-7012	4-2821-7159	301-2815-6868	0-2719-7265	1013.3± 1.2
AZ[10,SA]	2776-5-7203	9413-28-543	7150-139-2695	7211-292-2481	2737-235-7012	—	0-24-9960	0-0-9984	0-0-9984	1104.6 ± 1.2
AZ[10,QA]	2598-0-7386	9393-0-591	6976-0-3008	7065-0-2919	2821-4-7159	24-0-9960	—	0-0-9984	0-0-9984	1106.0 ± 1.2
AZ[100,SA]	2907-11-7066	9486-46-452	7299-174-2511	7238-366-2380	2815-301-6868	0-0-9984	0-0-9984	—	0-0-9984	1106.1 ± 1.2
AZ[100,QA]	2625-0-7359	9493-0-491	7453-0-2531	7524-0-2460	2719-0-7265	0-0-9984	0-0-9984	0-0-9984	—	1110.6 ± 1.2

Table 4: Tournament results for the Barbell graph. Each cell shows W-L-D (Wins-Losses-Draws) from the perspective of the row player against the column player. MCTS[100,SA] was used as the Elo = 1000.0 baseline.

4,992 games with the first agent as blue, for a total of (9 choose 2) * 4,992 * 2 = 359,424 games played. We evaluated Elo scores and their errors using the process described in Appendix A. The results are shown in Table 4.

The results are displayed so that agents with all parameters identical except for whether they are type (a) (having internal access to QC) or type (b) (not having such access – see Fig. 2) are shown in adjacent rows separated by lines. Comparing the four cases we tried gives Elo differences of $+13.6\pm1.7$, $+13.3\pm1.2$, $+1.4\pm1.7$, and = $+4.5\pm1.7$ for MCTS[10], MCTS[100], AZ[10], and AZ[100] respectively, all favoring agents with internal access to QC. These advantages translate to increases in expected mean win rate $E_w - E_w^c$ of 2.0 $\pm$ 0.2%, 1.9 $\pm$ 0.2%, $0.2 \pm 0.2\%$, and $0.6 \pm 0.2\%$ (recall that if all terminal states were visited equally likely by these agents the expected increase in mean win rate was 0.9%).

3-Prism

The Tangled instance $T(Y_3, (0, 4), 0.125, 100\text{ns})$, where Y_3 is the 3-Prism graph, contains $2,175$ out of $19,683$ ($\sim 11.1\%$) terminal states where simulated annealing corrupts reward. Of these, 75 give the win to the wrong player, and $2,100$ are terminal states where one of the judges rules a draw and the other rules a win for one of the two players (Table 2), giving an estimated increase in mean win rates assuming equal visitation of terminal states of $E_w - E_w^c \sim 75/19,683 + 0.5 * 2,100/19,683 \sim 5.7\%$.

We repeated the round-robin tournament using the same format as used for the Barbell graph. The results are shown in Table 5. Elo score mean differences are now $+33.3 \pm 1.7$, $+25.8 \pm 1.2$, $+37.5 \pm 1.7$, and $+49.0 \pm 1.7$ for MCTS[10], MCTS[100], AZ[10], and AZ[100] respectively, all favoring agents with internal access to QC. These advantages translate to expected mean increases in $E_w - E_w^c$ of $4.8 \pm 0.2\%$, $3.7 \pm 0.2\%$, $5.4 \pm 0.2\%$, and $7.0 \pm 0.2\%$.

Player	**MCTS[100,SA]**	**Random**	**MCTS[10,SA]**	**MCTS[10,QA]**	**MCTS[100,QA]**	**AZ[10,SA]**	**AZ[10,QA]**	**AZ[100,SA]**	**AZ[100,QA]**	**Elo**
Random	35-7848-2101	—	403-5524-4057	363-6146-3475	5-8720-1259	6-8587-1391	0-8836-1148	6-8868-1110	0-9137-847	598.0 ± 1.2
MCTS[10,SA]	116-4814-5054	5524-403-4057	—	1703-2327-5954	55-4990-4939	35-5491-4458	0-6409-3575	16-6489-3479	0-6666-3318	815.3 ± 1.2
MCTS[10,QA]	210-3434-6340	6146-363-3475	2327-1703-5954	—	68-4789-5127	58-4797-5129	0-5818-4166	50-5501-4433	0-6246-3738	848.6 ± 1.2
MCTS[100,SA]	—	7848-35-2101	4814-116-5054	3434-210-6340	542-947-8495	31-1221-8732	14-1750-8220	54-1835-8095	0-1960-8024	1000.0
MCTS[100,QA]	947-542-8495	8720-5-1259	4990-55-4939	4789-68-5127	—	436-834-8714	44-1604-8336	602-1181-8201	35-2038-7911	1025.8 ± 1.2
AZ[10,SA]	1221-31-8732	8587-6-1391	5491-35-4458	4797-58-5129	834-436-8714	—	0-127-9857	0-0-9984	0-2429-7555	1041.8 ± 1.2
AZ[10,QA]	1750-14-8220	8836-0-1148	6409-0-3575	5818-0-4166	1604-44-8336	127-0-9857	—	1487-0-8497	0-68-9916	1079.2 ± 1.2
AZ[100,SA]	1835-54-8095	8868-6-1110	6489-16-3479	5501-50-4433	1181-602-8201	0-0-9984	0-1487-8497	—	0-1744-8240	1051.1 ± 1.2
AZ[100,QA]	1960-0-8024	9137-0-847	6666-0-3318	6246-0-3738	2038-35-7911	2429-0-7555	68-0-9916	1744-0-8240	—	1100.1 ± 1.2

Table 5: Tournament results for the 3-Prism graph. Each cell shows W-L-D (Wins-Losses-Draws) from the perspective of the row player against the column player. MCTS[100,SA] was used as the Elo = 1000.0 baseline.

Moser Spindle

The Tangled instance $T(M_7, (2,4), 0.05, 40\text{ns})$, where M_7 is the Moser Spindle graph, contains $43,965$ out of $177,147$ ($\sim 24.8\%$) terminal states where simulated annealing corrupts reward (see Table 3), giving an estimated increase in mean win rates assuming equal visitation of terminal states of $E_w - E_w^c \sim 4,442/177,147 + 0.5 * 39,523/177,147 \sim 13.7\%$.

We repeated the round-robin tournament using the same format as used for the Barbell and 3-Prism graphs. The results are shown in Table 6. Elo score mean differences are $+79.1 \pm 1.7$, $+95.2 \pm 1.2$, $+98.4 \pm 1.7$, and $+108.4 \pm 1.7$ for MCTS[10], MCTS[100], AZ[10], and AZ[100] respectively, in each case favoring agents with internal access to QC. These advantages translate to expected mean increases in $E_w - E_w^c$ of $11.2 \pm 0.2\%$, $13.4 \pm 0.2\%$, $13.8 \pm 0.2\%$, and $15.1 \pm 0.2\%$.

Analysis of Results

For the graph instances studied here, we find the following results:

- In all cases, agents learning from true reward outperformed agents learning from corrupted reward. The magnitude of the effect is consistent with roughly equal visitation of terminal states by the agents. While we used Simulated Annealing as a specific classical approximation algorithm, we expect the findings here to hold for any algorithm that deterministically corrupts reward.
- Even in the case where the reward corruption was relatively high (for the Moser Spindle), agents learning from corrupted reward dramatically outperformed the random agent. This may be a consequence of the high percentage of mis-specification errors mis-labeling draws, which are less dangerous than errors assigning the win to the wrong player. This may mean that taking the draw boundary ϵ to zero, thereby removing draws from the game, may make the effects of reward corruption much worse.
- With $\epsilon = 0$, the No Free Lunch result [16] indicates that reward corruption of more than 50% of terminal states, given no further information, bounds agents learning from corrupted reward to be no better than the random agent. It seems plausible we can find such cases.

Player	**MCTS[100,SA]**	**Random**	**MCTS[10,SA]**	**MCTS[10,QA]**	**MCTS[100,QA]**	**AZ[10,SA]**	**AZ[10,QA]**	**AZ[100,SA]**	**AZ[100,QA]**	**Elo**
Random	207-7549-2228	—	967-6103-2914	725-7307-1952	26-9370-588	27-8369-1588	6-9252-726	27-8502-1455	4-9543-437	620.4 ± 1.2
MCTS[10,SA]	561-5991-3432	6103-967-2914	—	2186-3840-3958	293-7324-2367	144-7794-2046	42-8374-1568	86-8025-1873	22-8982-980	793.9 ± 1.2
MCTS[10,QA]	1025-3628-5331	7307-725-1952	3840-2186-3958	—	418-7087-2479	212-5817-3955	93-7814-2077	142-5973-3869	22-8850-1112	873.0 ± 1.2
MCTS[100,SA]	—	7549-207-2228	5991-561-3432	3628-1025-5331	1065-2964-5955	641-4478-4865	243-5018-4723	437-4887-4660	89-5805-4090	1000.0
MCTS[100,QA]	2964-1065-5955	9370-26-588	7324-293-2367	7087-418-2479	—	899-1666-7419	382-4385-5217	739-1815-7430	170-5616-4198	1095.2 ± 1.2
AZ[10,SA]	4478-641-4865	8369-27-1588	7794-144-2046	5817-212-3955	1666-899-7419	—	10-3086-6888	0-745-9239	0-2664-7320	1124.8 ± 1.2
AZ[10,QA]	5018-243-4723	9252-6-726	8374-42-1568	7814-93-2077	4385-382-5217	3086-10-6888	—	1278-0-8706	0-256-9728	1223.2 ± 1.2
AZ[100,SA]	4887-437-4660	8502-27-1455	8025-86-1873	5973-142-3869	1815-739-7430	745-0-9239	0-1278-8706	—	0-2734-7250	1149.9 ± 1.2
AZ[100,QA]	5805-89-4090	9543-4-437	8982-22-980	8850-22-1112	5616-170-4198	2664-0-7320	256-0-9728	2734-0-7250	—	1258.3 ± 1.2

Table 6: Tournament results for the Moser Spindle graph. Each cell shows W-L-D (Wins-Losses-Draws) from the perspective of the row player against the column player. MCTS[100,SA] was used as the Elo = 1000.0 baseline.

Pushing into the Advantage Regime: The C_{60} Fullerene Graph

The C_{60} Fullerene Graph, or Buckyball, is a 3-regular planar graph with $|V| = 60$ vertices and $|E| = 90$ edges, corresponding to the carbon structure of buckminsterfullerene [62]. This graph has been advanced as a potential testbed for quantum simulation of the sort relevant for Tangled, due to the interplay of degeneracy and quantum fluctuations of the sort that we saw defeat simulated annealing and may defeat other more powerful algorithms as well [41]. Other Fullerenes can be implemented if increased scale is required for quantum advantage, such as C_{70}, C_{84}, or C_{130}, which have all been observed in atmospheric aerosols [63].

As for C_{60}'s suitability as a human-playable game board, note that the state space is $4^{|E|} = 4^{90} \sim 10^{54}$, which is in the ballpark of chess ($\sim 10^{43}$) but much smaller than Go ($\sim 10^{170}$), and the number of moves per game ($|E| = 90$, 45 for each player) is also reasonably human-playable (chess has ~ 40, whereas Go has ~ 200). One could imagine playing on a physical bucky ball, where touching the edges changes their color; playing against an agent trained using a quantum computer that (allegedly) can't be beat might be a compelling experience.

We chose the Tangled instance $T(C_{60}, (0, 54), 0, 40\text{ns})$, where vertices 0 and 54 have distance 9, which is the diameter of this graph (see Fig. 13). Playing Tangled on this instance should be challenging for any classical approximation algorithm. Setting $\epsilon = 0$ makes the possible reward values $R \in \{+1, -1\}$ (no draws are allowed).

For this instance, generating a lookup table by enumerating terminal states is not an option (there are $3^{90} \sim 10^{43}$ of them). Instead, during the agent training phase, we evaluate the reward associated with terminal states and then discard it.

We trained two AlphaZero agents with identical parameters, except the first was trained on

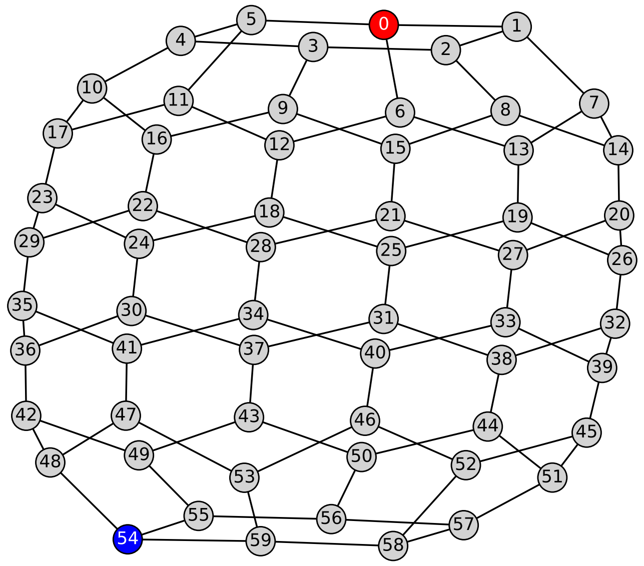


Figure 13: The 60-vertex 90-edge C_{60} Fullerene Graph, with player 1 vertex (vertex 0) colored red and player 2 vertex (vertex 54) colored blue.

Player	**Random**	**AZ[100, SA]**	**AZ[100, QA]**	**Elo**
Random	—	4-996	0-1000	1000
AZ[100, SA]	996-4	—	15-985	1718 ± 8
AZ[100, QA]	1000-0	985-15	—	2388 ± 8

Table 7: Tournament results for the C_{60} graph. Each cell shows W-L (Wins-Losses) from the perspective of the row player against the column player. The random agent was used as the Elo = 1000.0 baseline.

true reward computed using the D-Wave Advantage2.1.4 system, and the second was trained on corrupted reward (using simulated annealing to adjudicate terminal states). We used training parameters $M_0 = 100$, $M_1 = 10$, $N_{it} = 10$, $N_{eps} = 320$, $U = 1$, and $N_g = 320$. Note that training effort here was substantially lower than previous experiments due to time constraints. We are continuing to train agents on both true and corrupted reward over the course of the next three months.

We ran a round-robin tournament with both of them and a random agent, where each pair of agents played $N = 1,000$ games against each other, for a total of $3,000$ games played. The results are shown in Table 7.

The agent trained on true reward achieved a mean Elo difference of $+670 \pm 11$ over the agent trained on corrupted reward, which, if terminal states were visited equally, implies a reward corruption rate of $48 \pm 2\%$ – implying that in this regime simulated annealing assigns incorrect labels nearly half the time, close to the regime where its performance is expected to be no better than the random agent. Note that this is likely not the entire story, as the AlphaZero agent trained using corrupted reward trounces the random agent in 1-1 competition.

D. Classical Benchmarking

None of the current domains where quantum advantage has been claimed (including random circuit sampling [18], boson sampling [19], and D-Wave quantum simulation [20]) are universally accepted. In all cases, sophisticated classical algorithms may be able to reproduce the obtained results using reasonable resources.

This state of affairs will likely persist for some time. Our view is that eventually some domain will emerge where quantum advantage will be clear. It will likely be a type of quantum simulation, where the quantum computer experimentally demonstrates clear signatures of volume-law scaling of entanglement entropy [64], and the interesting properties of the system being simulated with the quantum computer critically depend on this (this should defeat approximation methods such as tensor networks that are limited to simulating systems where entanglement entropy has area-law scaling). Until such time, whenever any claim is made, researchers will target it and attempt to use whatever structure it has to devise clever special-purpose conventional algorithms to enable classical simulation and thereby try to refute the claim. Current quantum computational systems are just starting to be powerful enough to support reasonable claims, but they are not yet powerful enough to make these claims ironclad.

In the case of the D-Wave claim, the best reference to the current state of affairs is found here [44]; we refer the reader to this article for arguments and links to results.

E. Viability

The **Problem Statement** introduced at the beginning of this report is the following:

In the case where reward is the output of a quantum simulation, we aim to (a) discover what conditions are sufficient for RL agents with access to QC to be categorically superior to agents without such access, and (b) solve a set of industrially important challenge problems requiring design of quantum systems using such agents.

To succeed at (a), we need the following:

1. **Access to a quantum computing system that exhibits quantum advantage in the computation of the reward signal the RL agent uses.** In the work described here, there is a reasonable claim that this is true, but it is possible that the claim will be overturned. The general strategy outlined here can be applied to any quantum advantage claim domain.

2. **Reward corruption due to classically approximating the function where quantum advantage is claimed must lead to dramatically reduced agent performance.** Here we present empirical evidence that this is true for both MCTS and AlphaZero agents when the reward corruption is deterministic and there are no further simplifying assumptions on which states are corrupted. In addition, there are known information-theoretical results that are useful here [16].

3. **Agent performance upper bounds for error rates generated by running the best known classical algorithms must be lower than the performance achieved by an agent trained using true reward.** This type of bounding argument is required to rule out better learning algorithms learning from corrupted reward beating poorer learning algorithms learning from true reward. Computing and using such bounds is a central strategy in much of the quantum machine learning literature [12, 13, 14, 15], but deriving analogs for general quantum simulation is a much harder problem. For the simulation of a quantum system to be a domain of quantum advantage, its entanglement entropy should exhibit volume-law scaling to defeat methods that approximate entanglement as local, including tensor networks (this is at the heart of questions about whether the D-Wave quantum advantage claim will hold up). Whether such volume-law scaling is expected in a general quantum system is a difficult problem in practice, as most quantum systems of commercial interest are open quantum systems with complex environments, and the types of many-body quantum effects necessary to defeat classical methods may be rare.

To succeed at (b), success at (a) is a necessary condition. In addition, we must identify and solve a set of industrially relevant challenge problems where the following hold:

1. **Each problem in the challenge set has reward of the sort required for (a)**. This requires each quantum system identified in each challenge problem to have special properties that defeat specialized techniques for classical simulation of quantum systems, such as tensor networks, and these special properties must be directly connected to the ‘desirable properties’ encoded in reward. The reward must be computable in a domain of quantum advantage for an existing or near-term quantum computer.

2. **We must implement an agent, for example an AlphaZero agent, whose performance on each of these challenge problems empirically beats upper bounds for any agent trained using the best known classical approximations for the reward functions in question**. In Phase 1 of this project, we have implemented all of the infrastructure necessary to train AlphaZero agents given a general reward signal, and therefore should be (modulo lack of computing resources) prepared to implement such agents.

In Phase 1, we built the infrastructure necessary to design, implement, and test RL agents learning from arbitrary reward signals. We designed a special case where a specific type of quantum simulation is required to compute reward cast as win/loss/draw for a novel type of game. We applied our infrastructure to this game, and found that a reasonable classical approximation algorithm devastates an agent's ability to learn to play the game well. We identified at least one physical mechanism (order-by-disorder) that causes reward corruption in this case.

We then trained an AlphaZero agent on an instance of this game (on the C_{60} Fullerene Graph) in a regime proposed as a benchmark for quantum simulation. We demonstrated that an AlphaZero agent trained using simulated annealing for terminal state adjudication does no better than a random agent, and the AlphaZero agent trained on true reward soundly defeats both the random agent and the AlphaZero agent trained on corrupted reward.

These first-stage results are consistent with the project being viable, although there are many gaps between where we are at and being able to declare success. In the next phase, we will proceed to using the tools and infrastructure developed to date to identify and solve a set of industrially important challenge problems requiring design of quantum systems using RL agents.

F. Novelty

Quantum Circuits as General Function Approximators

The majority of work at the intersection of RL and QC proposes using quantum circuits as replacements for the neural nets used for learning in typical RL scenarios [13, 14, 15]. This approach has the same general goal as what we propose here – showing that when there exist conditions of quantum advantage, these can be translated to building RL agents that are categorically superior to agents with access to only classical computation.

Instead of replacing neural nets with quantum circuits as general function approximators, we propose keeping these neural nets as-is and focus quantum advantage specifically into computing reward. Theoretical results obtained by using quantum circuits in place of neural nets, for example for the case when computing reward requires solving the discrete log problem [12, 13], can be reframed as using those quantum circuits to instead compute reward. This cleanly separates learning from a reward signal and being able to compute it in the first place, which the approaches cited above conflate, while maintaining the theoretical separation guarantees obtained by previous authors.

Using RL to Design Quantum Systems

The closest work we were able to find to the work presented here involves training RL agents to design molecules with desired properties [65]. In this work, a very similar framework is built where the authors "... design molecules by sequentially drawing atoms from a given bag and placing them onto a 3D canvas... we focus on designing stable molecules, i.e. molecules with low energy $E \in R$; however, linear combinations of multiple desirable properties are possible as well". This is a special case of what we propose here, as the reward function is the result of a quantum simulation, and agents learn to design molecules subject to these rewards.

G. Evidence Supporting Section C (Quantum Advantage)

See Section D.

H. Evidence Supporting Section D (Classical Benchmarking)

See Section D.

Appendix A: Computing Elo Scores And Error Bounds

The Elo rating system, developed by Arpad Elo for chess skill assessment [45], provides a method for calculating relative skill levels in two-player zero-sum games. The system maintains numerical ratings for each player, updating them after each game based on the outcome and pre-game rating differences. The expected score for player i against player j is given by $E_i = 1/(1+10^{(R_j - R_i)/400})$, where R_i and R_j denote the current ratings. Following a game, ratings are updated according to $R_i^{\text{new}} = R_i + K(S_i - E_i)$, where $S_i \in \{0, 0.5, 1\}$ represents the actual game outcome (loss, draw, win) and K is a parameter controlling update magnitude, typically ranging from 10 to 40 (we used $K = 32$) [66]. The Elo system has found widespread application in diverse competitive domains including online gaming [67], sports analytics [68], and even academic journal rankings [69]. Despite its simplicity and popularity, the system has known limitations including lack of uncertainty quantification and game order dependence, motivating extensions such as Glicko [70] and TrueSkill [67].

Uncertainty Quantification for Elo Scores

Assume P players play a round-robin tournament, where each pair $(i, j), i < j$ plays N instances of a zero-sum game with draws allowed. Assume that the (unknown) true Elo score of player i is R_i^*. Furthermore, assume that pre-tournament, all players are assigned Elo scores of 1000, and assume player 0's Elo score is fixed to this initial value (Elo scores are relative to a baseline, so this can be done without loss of generality). Our goal is to choose N to be large enough so that the error bounds δR_i on the post-tournament Elo scores $R_i \pm \delta R_i$ are small enough to distinguish potentially small differences in capability between agents.

Specifically, we seek to find N such that the P players can be ordered by estimated post-tournament Elo scores R_i, where this is the same as the ordering of the true Elo scores R_i^*. If the difference between true Elo scores between players i and j is $\Delta R_{ij} = R_i^* - R_j^*$, we seek N such that $\delta R_i + \delta R_j < \Delta R_{ij} \, \forall \, (i, j)$.

Examining Each Pair

Assume the pair $(i, j), i < j$ play N games, with observed outcomes (from player i's perspective) of W wins, L losses, and D draws ($W + L + D = N$) and sample proportions $\hat{p}_w = W/N$, $\hat{p}_l = L/N$, and $\hat{p}_d = D/N$. These are estimators of the true probabilities p_w, p_l, and p_d.

Assuming the underlying probability distribution is multinomial, the standard errors for each estimate are $SE(\hat{p}_i) = \sqrt{p_i(1 - p_i)/N}$ where $i \in \{w, l, d\}$, which are all upper bounded by $SE(\hat{p}_i) = \sqrt{1/4N}$. We therefore know that the true probability estimates are $p_i = \hat{p}_i \pm z\sqrt{1/4N}$, where $z = 1.96$ provides a 95% confidence interval.

Assuming initial Elo scores of R_i and R_j, since Elo conserves the sum of ratings ($S_{ij} = R_i + R_j$), post-round Elo scores are:

$$R_i^{'} = S_{ij}/2 + 200 \log_{10}[(\hat{p}_w + 0.5\hat{p}_d)/(\hat{p}_l + 0.5\hat{p}_d)] \tag{15}$$

$$R_j^{'} = S_{ij}/2 - 200 \log_{10}[(\hat{p}_w + 0.5\hat{p}_d)/\hat{p}_l + 0.5\hat{p}_d)] \tag{16}$$

and the difference between estimated Elo scores is

$$\Delta R_{ij} = R_i^{'} - R_j^{'} = 400 \log_{10}[(\hat{p}_w + 0.5\hat{p}_d)/(\hat{p}_l + 0.5\hat{p}_d)] \tag{17}$$

Let

$$\hat{E}_w = \hat{p}_w + 0.5\hat{p}_d \tag{18}$$
$$\hat{E}_l = \hat{p}_l + 0.5\hat{p}_d = 1 - \hat{E}_w \tag{19}$$

then

$$\Delta R_{ij} = 400 \log_{10}[\hat{E}_w/(1 - \hat{E}_w)] \tag{20}$$

We are interested in the situation where ΔR_{ij} is small. This happens near $\hat{E}_w = 1/2$. Assume $\hat{E}_w = 1/2 + \epsilon$. Then

$$\Delta R_{ij} = 400 \log_{10}[(1/2 + \epsilon)/(1/2 - \epsilon)] \approx 400 \log_{10}[(1 + 4\epsilon)] \approx \frac{1600}{\ln(10)}\epsilon \tag{21}$$

and

$$\delta R_i = \delta R_j \approx \frac{800}{\ln(10)}\epsilon \tag{22}$$

To measure a small difference ΔR_{ij}, we need the errors in measuring $\hat{E}_w$ to be less than the ϵ required to give that value of ΔR_{ij}. Assuming $E_w = \hat{E}_w \pm z\sqrt{1/4N}$, then $z\sqrt{1/4N} < \epsilon$, and

$$\delta R_i = \delta R_j \approx \frac{400z}{\ln(10)\sqrt{N}} \tag{23}$$

To measure a difference of $\delta R_i = \delta R_j = 1 \Rightarrow \Delta R_{ij} = 2$, we need $N > 115,931$ games.

Our situation is complicated by the round-robin tournament format. There are methods for quantifying the computation of error bounds in this situation, such as using the Fisher Information Matrix [71]. The expectation is that the number of games each pair has to play to achieve the same error bound should be roughly reduced by a factor of $1/\sqrt{P-1}$ – so for a round-robin tournament with P players, the number of pairwise games required to achieve some Elo measurement threshold is reduced to $N \to N/\sqrt{P-1}$ pairwise games. We use this approximation here, and report

$$\delta R_i = \frac{400z}{\ln(10)\sqrt{N(P-1)}} \tag{24}$$

for the error on player i's Elo score, assuming a P player round-robin tournament where each player plays every other player N times.

Averaging Over Path Dependence

Assume two players have initial Elo scores of R_i and R_j for players i and j respectively. If these two players then play N games against each other, their post-play Elo scores depend on the order that the N game outcomes are processed. As a simple example, if $N = 2$ and each player wins one game and loses one game, the final Elo scores depend on the order we apply the results. To reduce this source of error in computing Elo, we can average over the final Elo scores for both orderings.

More generally, we estimate post-tournament Elo scores using the following approach. Given measured $\hat{p}_{ij,w} = W_{ij}/N$, $\hat{p}_{ij,l} = L_{ij}/N$, and $\hat{p}_{ij,d} = D_{ij}/N$, for play between players i and j where $i < j$, we create a random sequence of pairs (i, j) where each pair appears N times, and draw a sample from the multinomial distribution derived from the observed data for each of these, creating $NP(P-1)/2$ simulated game outcomes (in effect simulating one full round-robin tournament). We create M such sequences. For each sequence, we compute final Elo scores for all players by processing their games in order, starting from initial scores of 1000. After doing this for all M sequences, generating M final pairs of Elo scores R_p^m for $p = 0..P-1$ and $m = 0..M-1$, we compute the sample means $R_p =< R_p^m >_m$ and error bounds $\pm z < SE_p^m >_m /\sqrt{M}$. We choose M to be large enough so that $\frac{<SE_p^m>_m}{\sqrt{M}} << \frac{400}{\ln(10)\sqrt{N(P-1)}} \forall p$, our target Elo error due to finite sample size N. We found $M = 1,000$ gives $z < SE_p^m >_m /\sqrt{M} \sim 0.01$ for the N and P values used here, making its contribution to Elo uncertainty negligible.

We report the final Elo score for player i as

$$R_i =< R_i^m >_m \pm \frac{400z}{\ln(10)\sqrt{N(P-1)}} \tag{25}$$

which for $z = 1.96$, $N = 9,984$, and $P = 9$ gives

$$R_i =< R_i^m >_m \pm 1.2 \tag{26}$$

References

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, Second edition, 2018.

[2] David Silver, Satinder Singh, et al. Reward is enough. *Artificial Intelligence*, 299:103535, 2021.

[3] David Silver and Richard S. Sutton. Welcome to the Era of Experience. `https://storage.googleapis.com/deepmind-media/Era-of-Experience%20/The%20Era%20of%20Experience%20Paper.pdf`, 2025. To be published by MIT Press as a chapter in the book 'Designing an Intelligence', edited by George Konidaris.

[4] Richard S. Sutton, Michael Bowling, and Patrick M. Pilarski. The Alberta Plan for AI Research, 2022.

[5] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. IEEE, 1994.

[6] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[7] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 439(1907):553–558, 1992.

[8] Daniel R Simon. On the power of quantum computation. *SIAM journal on computing*, 26(5):1474–1483, 1997.

[9] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on computing*, 26(5):1411–1473, 1997.

[10] Seth Lloyd. Universal quantum simulators. *Science*, 273(5278):1073–1078, 1996.

[11] Dominic W Berry, Graeme Ahokas, Richard Cleve, and Barry C Sanders. Efficient quantum algorithms for simulating sparse hamiltonians. *Communications in Mathematical Physics*, 270(2):359–371, 2007.

[12] Yunchao Liu, Srinivasan Arunachalam, and Kristan Temme. A rigorous and robust quantum speed-up in supervised machine learning. *Nature Physics*, 17(9):1013–1017, 2021.

[13] Sofiene Jerbi, Casper Gyurik, Simon C. Marshall, Hans J. Briegel, and Vedran Dunjko. Parametrized quantum policies for reinforcement learning. *Advances in Neural Information Processing Systems*, 34:28362–28375, 2021. Demonstrates the first practical quantum-classical hybrid reinforcement learning model with provable quantum advantage for certain tasks.

[14] Andrea Skolik, Sofiene Jerbi, and Vedran Dunjko. Quantum agents in the gym: a variational quantum algorithm for deep q-learning. *Quantum*, 6:720, 2022. Introduces parametrized quantum circuits as Q-function approximators for deep reinforcement learning.

[15] Samuel Yen-Chi Chen, Chao-Han Huck Yang, Jun Qi, Pin-Yu Chen, Xiaoli Ma, and Hsi-Sheng Goan. Variational quantum circuits for deep reinforcement learning. *IEEE Access*, 8:141007–141024, 2020. First proof-of-principle demonstration of variational quantum circuits as deep Q-value function approximators.

[16] Tom Everitt, Victoria Krakovna, et al. Reinforcement learning with a corrupted reward channel. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)*, pages 4705–4713, 2017.

[17] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, 1982. Originally presented as a lecture at the Physics of Computation Conference, MIT Endicott House, May 1981.

[18] Frank Arute, Kunal Arya, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574:505–510, 2019.

[19] Han-Sen Zhong, Hui Wang, et al. Quantum computational advantage using photons. *Science*, 370(6523):1460–1463, 2020.

[20] Andrew D. King, Alberto Nocera, et al. Beyond-classical computation in quantum simulation. *Science*, 388(6743):199–204, 2025.

[21] Anton Korinek. The economics of transformative AI. *NBER Reporter*, 4, 2024.

[22] Kate Rooney. Tech megacaps plan to spend more than $300 billion in 2025 as AI race intensifies. CNBC, February 2025.

[23] Goldman Sachs Asset Management. Will the $1 trillion of generative AI investment pay off? Goldman Sachs Insights, August 2024.

[24] World Economic Forum. How venture capital is investing in AI in these 5 top economies. World Economic Forum, May 2024.

[25] Vikram Mittal and John Goetz. A quantitative analysis of the effects of drone and counter-drone systems on the Russia-Ukraine battlefield. *Defense & Security Analysis*, 2025.

[26] Hongming Chen, Ola Engkvist, et al. The rise of deep learning in drug discovery. *Drug discovery today*, 23(6):1241–1250, 2018.

[27] Jonathan M Stokes, Kevin Yang, et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702, 2020.

[28] Joseph A DiMasi, Henry G Grabowski, and Ronald W Hansen. Innovation in the pharmaceutical industry: new estimates of rd costs. *Journal of health economics*, 47:20–33, 2016.

[29] Zachary W Ulissi, Ankit R Singh, Charlie Tsai, and Jens K Nørskov. Machine learning methods in catalysis. *ACS Catalysis*, 7(10):6600–6608, 2017.

[30] Jonathan Schmidt, Mário RG Marques, Silvana Botti, and Miguel AL Marques. Recent advances and applications of machine learning in solid-state materials science. *npj Computational Materials*, 5(1):1–36, 2019.

[31] Muratahan Aykol, Patrick Herring, and Abe Anapolsky. Machine learning for continuous innovation in battery technologies. *Nature Reviews Materials*, 5(10):725–727, 2020.

[32] Connor W Coley, William H Green, and Klavs F Jensen. Machine learning in computer-aided synthesis planning. *Accounts of chemical research*, 51(5):1281–1289, 2018.

[33] James E Gubernatis and Turab Lookman. Machine learning in materials design and discovery: Examples from the present and suggestions for the future. *Physical Review Materials*, 2(12):120301, 2018.

[34] Lauri Himanen, Amber Geurts, et al. Data-driven materials science: status, challenges, and perspectives. *Advanced Science*, 6(21):1900808, 2019.

[35] Claude Shannon. Programming a computer for playing chess. *Philosophical Magazine*, 41(314), 1950.

[36] M. Campbell, A. J. Hoane Jr, and F. H. Hsu. Deep blue. *Artificial Intelligence*, 134:57–83, 2002.

[37] Cameron Browne, Edward Jack Powley, et al. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.

[38] David Silver, Aja Huang, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

[39] Christopher Berner, Greg Brockman, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[40] Oriol Vinyals, Igor Babuschkin, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, Nov 2019.

[41] Alejandro Lopez-Bezanilla, William Bernoudy, et al. Quantum dynamics in frustrated Ising fullerenes, May 2025. arXiv preprint.

[42] G. Rose. How I Adjudicate Tangled Terminal States Using D-Wave Hardware. `https://www.snowdropquantum.com/blog/how-i-solve-problems-using-d-wave-hardware`, 2024. Accessed August 2, 2025.

[43] Joseph Tindall, Antonio Mello, et al. Dynamics of disordered quantum systems with two- and three-dimensional tensor networks. *arXiv:2503.05693*, 2025.

[44] Andrew D. King, Alberto Nocera, et al. Comment on: "dynamics of disordered quantum systems with two- and three-dimensional tensor networks" arxiv:2503.05693. *arXiv preprint arXiv:2504.06283*, 2025.

[45] Arpad E. Elo. *The Rating of Chess Players, Past and Present*. Arco Publishing, first edition, 1978.

[46] Andrew Y Ng and Stuart J Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 663–670. Morgan Kaufmann Publishers Inc., 2000.

[47] Annealing schedules for the D-Wave Advantage2.1.3 System. `https://docs.dwavequantum.com/en/latest/quantum_research/solver_properties_specific.html#advantage2-system1-3`, 2025.

[48] J. Vannimenus and G. Toulouse. Theory of the frustration effect. II. Ising spins on a square lattice. *Journal of Physics C: Solid State Physics*, 10:L537, 1977.

[49] R. F. Wang, C. Nisoli, et al. Artificial 'spin ice' in a geometrically frustrated lattice of nanoscale ferromagnetic islands. *Nature*, 439:303–306, 2006.

[50] J. Villain, R. Bidaux, et al. Order as an effect of disorder. *Journal de Physique*, 41:1263–1272, 1980.

[51] R. Hanai. Nonreciprocal frustration: Time crystalline order-by-disorder phenomenon and a spin-glass-like state. *Physical Review X*, 14:011029, 2024.

[52] G. Schumm, H. Shao, et al. Primary and secondary order parameters in the fully frustrated transverse-field Ising model on the square lattice. *Physical Review B*, 109:L140408, 2024.

[53] J. D. M. Champion, M. J. Harris, et al. $Er_2ti_2o_7$: Evidence of quantum order by disorder in a frustrated antiferromagnet. *Physical Review B*, 68:020401, 2003.

[54] M. E. Zhitomirsky, M. V. Gvozdikova, et al. Quantum order by disorder and accidental soft mode in $er_2ti_2o_7$. *Physical Review Letters*, 109:077204, 2012.

[55] G. Rose. Tangled on a 3-vertex graph. `https://www.snowdropquantum.com/blog/tangled-on-a-three-vertex-graph`, Sept 2024.

[56] Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *Machine Learning: ECML 2006*, volume 4212 of *Lecture Notes in Computer Science*, pages 282–293. Springer, 2006.

[57] Maciej Świechowski, Konrad Godlewski, et al. Monte Carlo Tree Search: A review of recent modifications and applications. *Artificial Intelligence Review*, 2022.

[58] David Silver, Thomas Hubert, et al. A general reinforcement learning algorithm that masters Chess, Shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018.

[59] David Silver, Julian Schrittwieser, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[60] Christopher D. Rosin. Multi-armed bandits with episode context. *Annals of Mathematics and Artificial Intelligence*, 61(3):203–230, 2011.

[61] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778. IEEE, 2016.

[62] Harold W. Kroto, James R. Heath, Sean C. O'Brien, Richard F. Curl, and Richard E. Smalley. C_{60}: Buckminsterfullerene. *Nature*, 318:162–163, 1985.

[63] Fábio N. dos Santos, Madson M. Nascimento, Gisele O. da Rocha, and Jailson B. de Andrade. The occurrence of pristine and functionalized fullerenes as constituents of airborne aerosols. *Scientific Reports*, 13:4248, 2023.

[64] Eugenio Bianchi, Lucas Hackl, Mario Kieburg, Marcos Rigol, and Lev Vidmar. Volume-law entanglement entropy of typical pure quantum states. *PRX Quantum*, 3:030201, 2022.

[65] Gregor N. C. Simm, Robert Pinsler, and José Miguel Hernández-Lobato. Reinforcement learning for molecular design guided by quantum mechanics. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8959–8969. PMLR, 13–18 Jul 2020.

[66] Mark E. Glickman. A comprehensive guide to chess ratings. *American Chess Journal*, 3:59–102, 1995.

[67] Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill: a Bayesian skill rating system. In *Advances in Neural Information Processing Systems*, volume 19, pages 569–576. MIT Press, 2007.

[68] Lars Magnus Hvattum and Halvard Arntzen. Using ELO ratings for match result prediction in association football. *International Journal of Forecasting*, 26(3):460–470, 2010.

[69] Peter Vinkler. Would it be possible to increase the Elo rating of a journal by publishing more papers? *Journal of the Association for Information Science and Technology*, 64(4):787–799, 2013.

[70] Mark E. Glickman. Parameter estimation in large dynamic paired comparison experiments. *Applied Statistics*, 48(3):377–394, 1999.

[71] Daniel Gomes de Pinho Zanco, Leszek Szczecinski, et al. Stochastic analysis of the Elo rating algorithm in round-robin tournaments. *Digital Signal Processing*, 143:104086, 2023. Presents comprehensive stochastic analysis of Elo algorithm convergence characteristics.