Instance Space Analysis on Quantum Algorithms

Vivek Katial

20/07/2020

Introduction

About Me

- Vivek Katial (vkatial@student.unimelb.edu.au)
  - PhD Candidate in School of Mathematics and Statistics

Talk Structure

- Background
- Overview of Literature
- Current Progress
- What's next?

Main Research Question

- What instances characteristics of optimisation problems make them predisposed to being solved on a Quantum Computer?

Adiabatic Quantum Computing (AQC)

Adiabatic Quantum Computing (AQC)

- Adiabatic Quantum Computation is a computational model which relies on the *adiabatic thereom* of quantum mechanics to compute calculations.

Adiabatic Quantum Computing (AQC)

- Adiabatic Quantum Computation is a computational model which relies on the *adiabatic thereom* of quantum mechanics to compute calculations.
- The famous *Schrödinger Equation* is well-known to describe the time evolution of a quantum state:

Adiabatic Quantum Computing (AQC)

- Adiabatic Quantum Computation is a computational model which relies on the *adiabatic thereom* of quantum mechanics to compute calculations.
- The famous *Schrödinger Equation* is well-known to describe the time evolution of a quantum state:
- $$i\hbar \frac{d}{dt} |\psi(t)\rangle = H(t) |\psi(t)\rangle$$

Adiabatic Quantum Computing (AQC)

- Adiabatic Quantum Computation is a computational model which relies on the *adiabatic thereom* of quantum mechanics to compute calculations.
- The famous *Schrödinger Equation* is well-known to describe the time evolution of a quantum state:
- $$i\hbar\frac{d}{dt}|\psi(t)\rangle = H(t)|\psi(t)\rangle$$
- $|\psi(t)\rangle$ is our state vector, $H(t)$ is the time dependent Hamiltonian. A Hamiltonian of an n-qubit system $H(t)$ is given by $2^n \times 2^n$ matrix.

Adiabatic Quantum Computing (AQC)

- An adiabatic computation can be expressed by specifying two Hamiltonians, denoted by H_B and H_P where H_B is our *initial* Hamiltonian and H_P is the *final* or *problem* Hamiltonian.

Adiabatic Quantum Computing (AQC)

- An adiabatic computation can be expressed by specifying two Hamiltonians, denoted by H_B and H_P where H_B is our *initial* Hamiltonian and H_P is the *final* or *problem* Hamiltonian.
- $$\begin{array}{ccc} H_B & \xrightarrow{\hspace{4cm}} & H_P \\ |\psi_B\rangle & \text{transform } H(t) & |\psi_P\rangle \\ \text{known state} & & \text{unknown} \end{array}$$

Adiabatic Quantum Computing (AQC)

- An adiabatic computation can be expressed by specifying two Hamiltonians, denoted by H_B and H_P where H_B is our *initial* Hamiltonian and H_P is the *final* or *problem* Hamiltonian.
- $$\begin{array}{ccc} H_B & \xrightarrow{\qquad\qquad\qquad} & H_P \\ |\psi_B\rangle & \text{transform } H(t) & |\psi_P\rangle \\ \text{known state} & & \text{unknown} \end{array}$$
- Loosely speaking, the adiabatic theorem tells us that if we vary from H_B to H_P *slowly enough* the system will remain in its ground state. This fact is a direct result of the Adiabatic Theorem [1].

Adiabatic Quantum Computing (AQC)

- To conduct the computation we evolve $|\psi(t)\rangle$ till time $t = T$ such that $|\psi(t = T)\rangle$ encodes the answer. The computation is done using a Hamiltonian which interpolates between H_B and H_P. Specifically as below:

Adiabatic Quantum Computing (AQC)

- To conduct the computation we evolve $|\psi(t)\rangle$ till time $t = T$ such that $|\psi(t = T)\rangle$ encodes the answer. The computation is done using a Hamiltonian which interpolates between H_B and H_P. Specifically as below:
- $$H(t) = [1 - \lambda(t)]\, H_B + \lambda(t) H_P, \qquad \lambda(t) = \frac{t}{T}$$

Adiabatic Quantum Computing (AQC)

- To conduct the computation we evolve $|\psi(t)\rangle$ till time $t = T$ such that $|\psi(t = T)\rangle$ encodes the answer. The computation is done using a Hamiltonian which interpolates between H_B and H_P. Specifically as below:
- $$H(t) = [1 - \lambda(t)]\, H_B + \lambda(t) H_P, \qquad \lambda(t) = \frac{t}{T}$$
- How fast?

Adiabatic Quantum Computing (AQC)

- To conduct the computation we evolve $|\psi(t)\rangle$ till time $t = T$ such that $|\psi(t = T)\rangle$ encodes the answer. The computation is done using a Hamiltonian which interpolates between H_B and H_P. Specifically as below:
- $$H(t) = [1 - \lambda(t)]\, H_B + \lambda(t) H_P, \qquad \lambda(t) = \frac{t}{T}$$
- How fast?
- $$T = \frac{1}{\min_t g(t)^2}$$

Adiabatic Quantum Computing (AQC)

- To conduct the computation we evolve $|\psi(t)\rangle$ till time $t = T$ such that $|\psi(t = T)\rangle$ encodes the answer. The computation is done using a Hamiltonian which interpolates between H_B and H_P. Specifically as below:
- $$H(t) = [1 - \lambda(t)]\, H_B + \lambda(t) H_P, \qquad \lambda(t) = \frac{t}{T}$$
- How fast?
- $$T = \frac{1}{\min_t g(t)^2}$$
- Where $g(t)$ is the difference between the first two smallest eigenvalues of $H(t)$

3SAT - Exact Cover

3SAT (Exact Cover)

- The satisfiability problem, abbreviated SAT, is a classic example of an NP-complete problem [2]

3SAT (Exact Cover)

- The satisfiability problem, abbreviated SAT, is a classic example of an NP-complete problem [2]
- The basic SAT formulation can be described as follows: Given a boolean formula (AND $\wedge$, OR $\vee$, NOT $\neg$) over n variables $(z_1, z_2, ..., z_n)$. Can one set z_i's in a manner such that the Boolean formula is true?

3SAT (Exact Cover)

- The satisfiability problem, abbreviated SAT, is a classic example of an NP-complete problem [2]
- The basic SAT formulation can be described as follows: Given a boolean formula (AND $\wedge$, OR $\vee$, NOT $\neg$) over n variables $(z_1, z_2, ..., z_n)$. Can one set z_i's in a manner such that the Boolean formula is true?
- A clause is an expression which the variables must satisfy. For example $z_1 \wedge z_2 \implies z_1 = z_2 = 1$

3SAT (Exact Cover Example)

- Consider a 4-bit number with two clauses:

3SAT (Exact Cover Example)

- Consider a 4-bit number with two clauses:
  - $z_1 \wedge z_2 \wedge z_3$

3SAT (Exact Cover Example)

- Consider a 4-bit number with two clauses:
  - $z_1 \wedge z_2 \wedge z_3$
  - $z_1 \wedge z_3 \wedge z_4$

3SAT (Exact Cover Example)

- Consider a 4-bit number with two clauses:
  - $z_1 \wedge z_2 \wedge z_3$
  - $z_1 \wedge z_3 \wedge z_4$
- Here we have 16 possible assignments, namely:

3SAT (Exact Cover Example)

- Consider a 4-bit number with two clauses:
    - $z_1 \wedge z_2 \wedge z_3$
    - $z_1 \wedge z_3 \wedge z_4$
- Here we have 16 possible assignments, namely:
-

$$\mathcal{Z} = \{\, 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111,\\ 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111 \,\}$$

3SAT (Exact Cover Example)

- Consider a 4-bit number with two clauses:
  - $z_1 \wedge z_2 \wedge z_3$
  - $z_1 \wedge z_3 \wedge z_4$
- Here we have 16 possible assignments, namely:
-

$$\mathcal{Z} = \{ 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, \\ 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111 \}$$

- However, our **satisfying assignment** is only $\vec{z} = 1111$

3SAT (Exact Cover Example)

- Consider a 4-bit number with two clauses:
  - $z_1 \wedge z_2 \wedge z_3$
  - $z_1 \wedge z_3 \wedge z_4$
- Here we have 16 possible assignments, namely:
-

$$\mathcal{Z} = \{\, 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, \\ 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111 \,\}$$

- However, our **satisfying assignment** is only $\vec{z} = 1111$
- We call a 3SAT problem with one satisfying assignment an instance of USA

3SAT (Exact Cover Example)

- Consider a 4-bit number with two clauses:
  - $z_1 \wedge z_2 \wedge z_3$
  - $z_1 \wedge z_3 \wedge z_4$
- Here we have 16 possible assignments, namely:
-

$$\mathcal{Z} = \{\, 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, \\ 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111 \,\}$$

- However, our **satisfying assignment** is only $\vec{z} = 1111$
- We call a 3SAT problem with one satisfying assignment an instance of USA
- Exact Cover implies that clauses are "exclusive" and in the form of $z_i + z_j + z_k = 1$

Mapping 3SAT to AQC

- How do we construct H_B and H_P so that we can solve our optimisation problem?

Mapping 3SAT to AQC

- How do we construct H_B and H_P so that we can solve our optimisation problem?
- Farhi et al.[3] describe this construction in considerable detail.

Mapping 3SAT to AQC

- How do we construct H_B and H_P so that we can solve our optimisation problem?
- Farhi et al.[3] describe this construction in considerable detail.
- We then evolve our system with the following interpolation:

Mapping 3SAT to AQC

- How do we construct H_B and H_P so that we can solve our optimisation problem?
- Farhi et al.[3] describe this construction in considerable detail.
- We then evolve our system with the following interpolation:
- $$H(t) = [1 - \lambda(t)]\, H_B + \lambda(t) H_P, \qquad \lambda(t) = \frac{t}{T}$$

Mapping 3SAT to AQC

- How do we construct H_B and H_P so that we can solve our optimisation problem?
- Farhi et al.[3] describe this construction in considerable detail.
- We then evolve our system with the following interpolation:
- $$H(t) = [1 - \lambda(t)]\, H_B + \lambda(t) H_P, \qquad \lambda(t) = \frac{t}{T}$$
- Finally, complete measurement and our solution is our final state $|\psi(t = T)\rangle$

Instance Space Analysis

Algorithm Selection

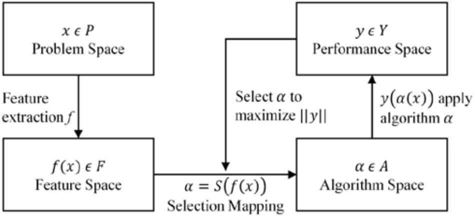
- Given a set of problem instances, predicting which algorithm is most likely to best perform was first explored by Rice [4].

Algorithm Selection

- Given a set of problem instances, predicting which algorithm is most likely to best perform was first explored by Rice [4].

Algorithm Selection

- Given a set of problem instances, predicting which algorithm is most likely to best perform was first explored by Rice [4].



- However, what we are interested in is probing the strengths and weaknesses of AQC for different instances of SAT.

Instance Space Methodology

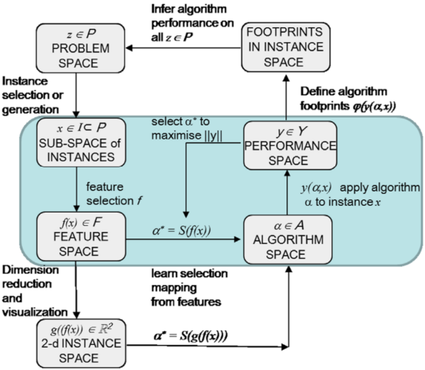
- The instance space methodology presented in [5–7] extends Rice's framework.

Instance Space Methodology

- The instance space methodology presented in [5–7] extends Rice's framework.

Instance Space Methodology

- The instance space methodology presented in [5–7] extends Rice's framework.



Current Literature

Quantum Computing Research into Hard SAT problems

- In their seminal paper [3] Farhi et al. aswell as Hogg et al. [8] touched on the phase transition and showed that the minimum energy gap perhaps scaled polynomially with n as roughly N^2.

Quantum Computing Research into Hard SAT problems

- In their seminal paper [3] Farhi et al. aswell as Hogg et al. [8] touched on the phase transition and showed that the minimum energy gap perhaps scaled polynomially with n as roughly N^2.
- However in 2009, Young et al [9]. demonstrated via Quantum monte-carlo simulations of $N = 256$ that some USA instances have a *quantum phase transition*.

Quantum Computing Research into Hard SAT problems

- In their seminal paper [3] Farhi et al. aswell as Hogg et al. [8] touched on the phase transition and showed that the minimum energy gap perhaps scaled polynomially with n as roughly N^2.
- However in 2009, Young et al [9]. demonstrated via Quantum monte-carlo simulations of $N = 256$ that some USA instances have a *quantum phase transition*.
- Their research indicated that as $N \rightarrow \infty$ the system is expected to lead to an exponentially small gap, and hence an exponential complexity.

Quantum Computing Research into Hard SAT problems

- In their seminal paper [3] Farhi et al. aswell as Hogg et al. [8] touched on the phase transition and showed that the minimum energy gap perhaps scaled polynomially with n as roughly N^2.
- However in 2009, Young et al [9]. demonstrated via Quantum monte-carlo simulations of $N = 256$ that some USA instances have a *quantum phase transition*.
- Their research indicated that as $N \to \infty$ the system is expected to lead to an exponentially small gap, and hence an exponential complexity.
- Farhi et al. [10] also investigated different evolution paths and their results suggested it is possible to overcome the exponentially small minimum gap by selecting random initial Hamiltonians

Quantum Computing Research into Hard SAT problems

- Latorre et al. probed the entropy of entanglement for 250 USA Instances of Exact-Cover [11], their results showed that entropy of entanglement scales linearly with N.

Quantum Computing Research into Hard SAT problems

- Latorre et al. probed the entropy of entanglement for 250 USA Instances of Exact-Cover [11], their results showed that entropy of entanglement scales linearly with N.
- However, their results also showed that for large N, the minimum gap scaling is exponential and can be associated with a *quantum phase transition* [11].

Quantum Computing Research into Hard SAT problems

- Latorre et al. probed the entropy of entanglement for 250 USA Instances of Exact-Cover [11], their results showed that entropy of entanglement scales linearly with N.
- However, their results also showed that for large N, the minimum gap scaling is exponential and can be associated with a *quantum phase transition* [11].
- Hauke et al. [12] ran simulations of adiabatic quantum optimisation with $n = 16$. Their results indicated that large entanglement entropy has little significance for the success probability of the optimisation task.

Quantum Computing Research into Hard SAT problems (D-Wave)

- Mapping optimisation problems as QUBO problems unlocks a significant number of NP-complete problems to investigate [13]

Quantum Computing Research into Hard SAT problems (D-Wave)

- Mapping optimisation problems as QUBO problems unlocks a significant number of NP-complete problems to investigate [13]
- Katzgrabber et al. [14] found Quantum annealing performed slightly better than classical ML algorithms on a computational biology problem

Quantum Computing Research into Hard SAT problems (D-Wave)

- Mapping optimisation problems as QUBO problems unlocks a significant number of NP-complete problems to investigate [13]
- Katzgrabber et al. [14] found Quantum annealing performed slightly better than classical ML algorithms on a computational biology problem
- Gabor et al. [15] shows that the phase transition from 3SAT persists in some form (but possibly to a lesser extent) in AQC via **real experimental results** on D-WAVE.

Quantum Computing Research into Hard SAT problems (D-Wave)

- Mapping optimisation problems as QUBO problems unlocks a significant number of NP-complete problems to investigate [13]
- Katzgrabber et al. [14] found Quantum annealing performed slightly better than classical ML algorithms on a computational biology problem
- Gabor et al. [15] shows that the phase transition from 3SAT persists in some form (but possibly to a lesser extent) in AQC via **real experimental results** on D-WAVE.
- Other promising studies in election forecasting and human cancer detection [16].

Quantum Computing Research into Hard SAT problems (D-Wave)

- Mapping optimisation problems as QUBO problems unlocks a significant number of NP-complete problems to investigate [13]
- Katzgrabber et al. [14] found Quantum annealing performed slightly better than classical ML algorithms on a computational biology problem
- Gabor et al. [15] shows that the phase transition from 3SAT persists in some form (but possibly to a lesser extent) in AQC via **real experimental results** on D-WAVE.
- Other promising studies in election forecasting and human cancer detection [16].
- Current approaches fail to investigate a suitable class of instances.

Research Overview

Research Overview

- Depending on different structures of instances, classical algorithms may perform differently to Quantum ones.

Research Overview

- Depending on different structures of instances, classical algorithms may perform differently to Quantum ones.
- We are looking to investigate which types of instances are more pre-disposed to being solved on Quantum Computers. Currently, we have explored two types of instances:

Research Overview

- Depending on different structures of instances, classical algorithms may perform differently to Quantum ones.
- We are looking to investigate which types of instances are more pre-disposed to being solved on Quantum Computers. Currently, we have explored two types of instances:
    - Relaxed USA Instances

Research Overview

- Depending on different structures of instances, classical algorithms may perform differently to Quantum ones.
- We are looking to investigate which types of instances are more pre-disposed to being solved on Quantum Computers. Currently, we have explored two types of instances:
  - Relaxed USA Instances
  - Generalised USA Instances

Research Overview

- We then have simulated AQC and probe the “quantumness” of each instance by measuring:

Research Overview

- We then have simulated AQC and probe the "quantumness" of each instance by measuring:
    1. Minimum Energy Gap g_{min}

Research Overview

- We then have simulated AQC and probe the "quantumness" of each instance by measuring:
    1. Minimum Energy Gap g_{min}
    2. Bipartite Entropy of Entanglement

Research Overview

- We then have simulated AQC and probe the “quantumness” of each instance by measuring:
  1. Minimum Energy Gap g_{min}
  2. Bipartite Entropy of Entanglement
  3. Probability of sucess after a fixed run time T

Research Overview - Instance Space

- The problem space $\mathcal{P}$ consists of all possible 3SAT instances of Exact Cover.

Research Overview - Instance Space

- The problem space $\mathcal{P}$ consists of all possible 3SAT instances of Exact Cover.
- The instance space $\mathcal{I} \subset \mathcal{P}$ compromises of 5760 RUSA and GUSA instances. These range from 5 to 11 qubits and are randomly generated.

Research Overview - Instance Space

- The problem space $\mathcal{P}$ consists of all possible 3SAT instances of Exact Cover.
- The instance space $\mathcal{I} \subset \mathcal{P}$ compromises of 5760 RUSA and GUSA instances. These range from 5 to 11 qubits and are randomly generated.
- The feature space $\mathcal{F}$ consists of 23 different instance features generated

Research Overview - Instance Space

- The problem space $\mathcal{P}$ consists of all possible 3SAT instances of Exact Cover.
- The instance space $\mathcal{I} \subset \mathcal{P}$ compromises of 5760 RUSA and GUSA instances. These range from 5 to 11 qubits and are randomly generated.
- The feature space $\mathcal{F}$ consists of 23 different instance features generated
- The algorithm portfolio $\mathcal{A}$ includes 16 algorithm parameter configurations for the run time T and also the time step Δt. We also are using a single path function $\lambda(t)$

Research Overview - Instance Space

- The problem space $\mathcal{P}$ consists of all possible 3SAT instances of Exact Cover.
- The instance space $\mathcal{I} \subset \mathcal{P}$ compromises of 5760 RUSA and GUSA instances. These range from 5 to 11 qubits and are randomly generated.
- The feature space $\mathcal{F}$ consists of 23 different instance features generated
- The algorithm portfolio $\mathcal{A}$ includes 16 algorithm parameter configurations for the run time T and also the time step Δt. We also are using a single path function $\lambda(t)$
- The performance metric $y \in \mathcal{Y}$ is the probability of success for the algorithm.

Research Overview - Generating GUSA Instances

Algorithm 1: Generalised USA Instances

Fix number of bits to n;
$C = \{\}$;
$i = 0$;
while *While number of satisfying assignments* > 0 **do**
    $C_i =$ Three distinct bits randomly from a uniform distribution;
    $i = i + 1$;
    if *number of satisfying assignments* $= 1$ **then**
        return $(n, \mathbf{C})$;
    else if *number of satisfying assignments = 0* **then**
        restart;
    else if *number of satisfying assignments has decreased* **then**
        Add C_i into $\mathbf{C}$;
end
Result: $(n, \mathbf{C})$: n variables with a set of clauses $\mathbf{C}$

Research Overview - Generating RUSA Instances

Algorithm 2: Relaxed USA Instances

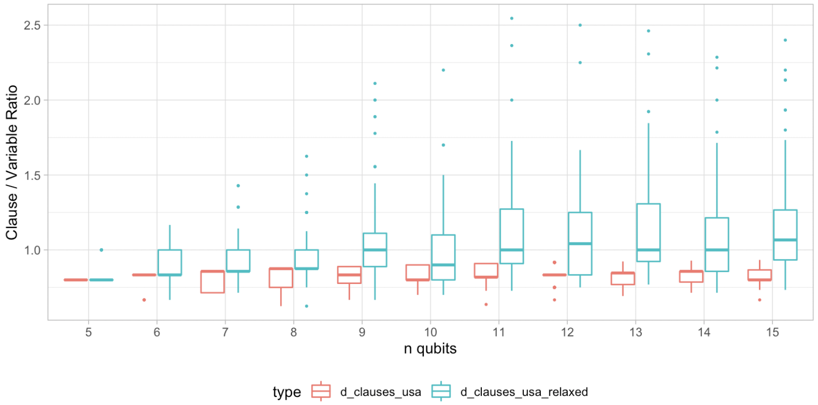
Fix number of bits to n;
$C = \{\}$;
$i = 0$;
while *While number of satisfying assignments* > 0 **do**
    C_i = Three distinct bits randomly from a uniform distribution;
    $i = i + 1$;
    if *number of satisfying assignments* $= 1$ **then**
        return $(n, \mathbf{C})$;
    else if *number of satisfying assignments = 0* **then**
        restart;
    else
        Add C_i into $\mathbf{C}$;
    end
end
Result: $(n, \mathbf{C})$: n variables with a set of clauses $\mathbf{C}$

Research Overview - $f(x)$ Instance Characterstics

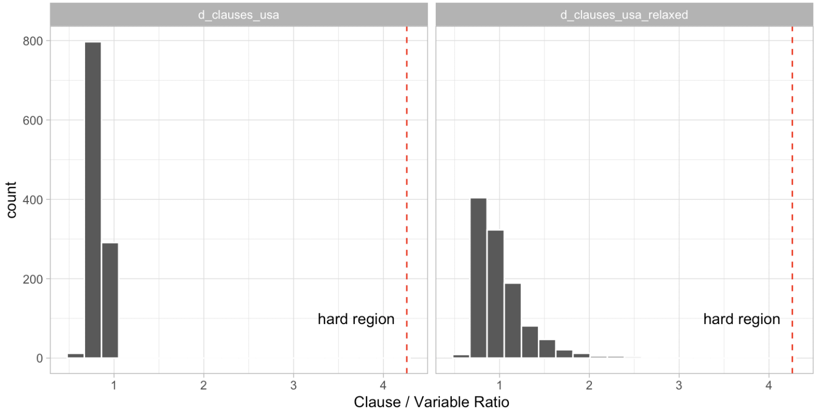
Feature Group	Feature
Problem Size	Number of variables: n Number of clauses: m Clause-to-Variable Ratio: $\frac{n}{m}$, $\frac{n}{m}^2$,$\frac{n}{m}^3$ Inverse Clause-to-Variable Ratio: $\frac{m}{n}$, $\frac{m}{n}^2$,$\frac{m}{n}^3$ Linearised Clause to Variable Ratio: $\|4.26 - \frac{n}{m}\|, \|4.26 - \frac{n}{m}\|^2, \|4.26 - \frac{n}{m}\|^3$
Variable Clause Graph	Variable Node Degree: mean, median, min, max Clause Node Degree : mean, median, min, max
Variable Graph	Node Degree : mean, median, min, max

Table 1: Instance Features for 3SAT Exact Cover

Research Overview - Distribution of Features

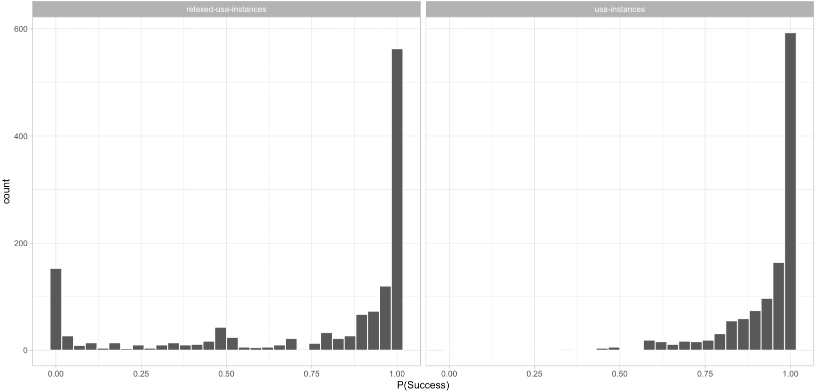


Research Overview - Distribution of Features

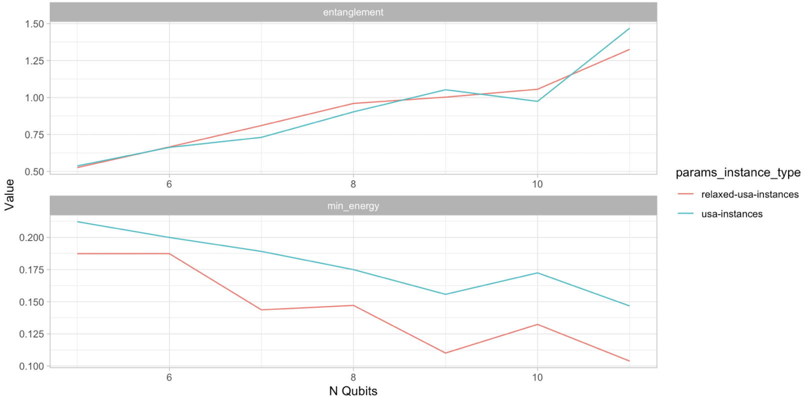


Results

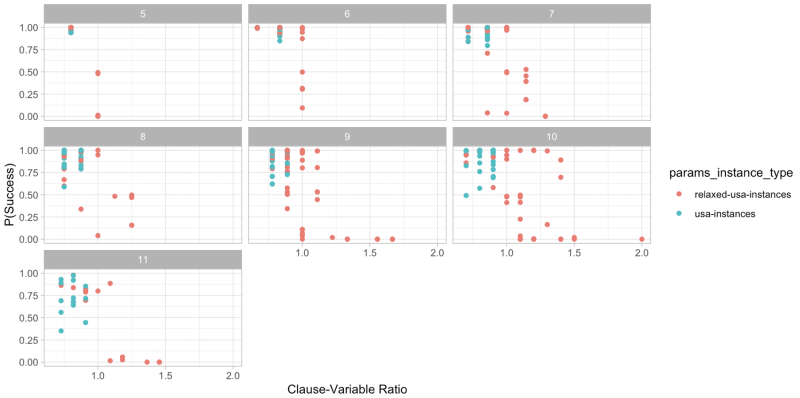
Research Overview - Results



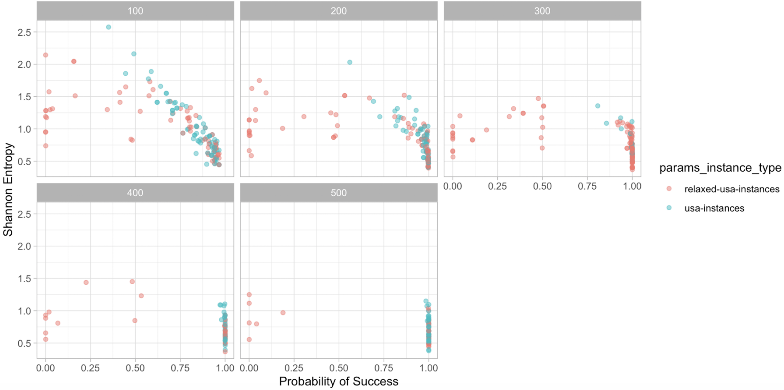
Research Overview - Results



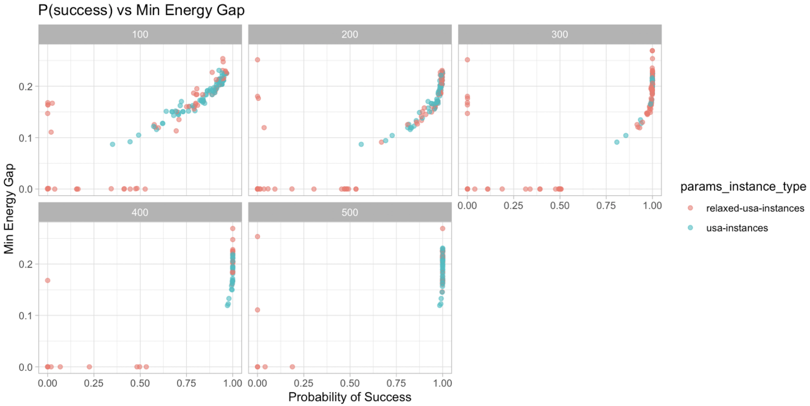
Research Overview - Results



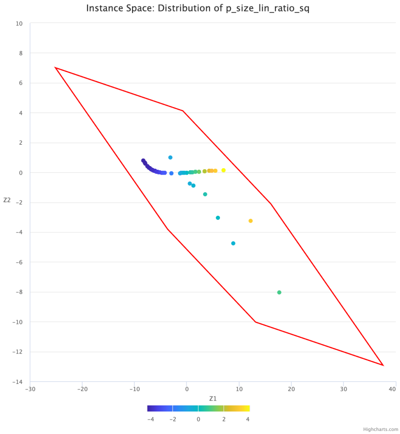
Research Overview - Entropy



Research Overview - Minimum Energy Gap

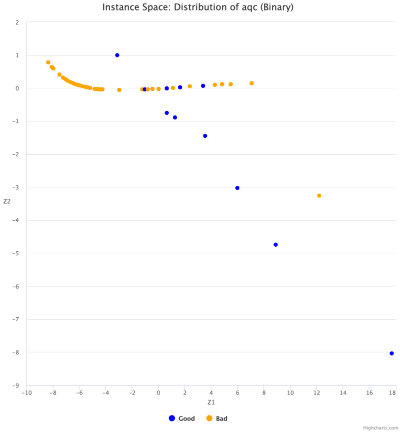


Research Overview - Results



Research Overview - Results

- We set a threshold $\tau = 0.95$ for “good” or easy instances.



Next Steps

Next Steps

1. Generate more instances based on MATILDA results

Next Steps

1. Generate more instances based on MATILDA results
2. Implement more performance metrics on experiments

Next Steps

1. Generate more instances based on MATILDA results
2. Implement more performance metrics on experiments
3. Identify which time step Δt is most suitable for evolution

Next Steps

1. Generate more instances based on MATILDA results
2. Implement more performance metrics on experiments
3. Identify which time step Δt is most suitable for evolution
4. Investigate the effects of randomising initial Hamiltonians

Next Steps

5. Explore QMC and Matrix-Product States to elicit further insights about AQC

Next Steps

5. Explore QMC and Matrix-Product States to elicit further insights about AQC
6. Extend implementation to work on QAOA and VQE with `QisKit`

Next Steps

5. Explore QMC and Matrix-Product States to elicit further insights about AQC
6. Extend implementation to work on QAOA and VQE with `QisKit`
7. Apply ISA to results from QAOA and run QAOA on Universal Quantum Computer

Next Steps

5. Explore QMC and Matrix-Product States to elicit further insights about AQC
6. Extend implementation to work on QAOA and VQE with `QisKit`
7. Apply ISA to results from QAOA and run QAOA on Universal Quantum Computer
8. Extend research to look at other QUBO problems

References

References I

1. Born M, Fock V (1928) Beweis des adiabatensatzes. *Zeitschrift für Physik* 51: 165–180.

2. Cook SA (1971) The complexity of theorem-proving procedures, In, *Proceedings of the third annual acm symposium on theory of computing*, ACM, 151–158.

3. Farhi E, Goldstone J, Gutmann S, et al. (2001) A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem. *Science* 292: 472–475.

4. Rice JR, others (1976) The algorithm selection problem. *Advances in computers* 15: 5.

5. Smith-Miles K, Bowly S (2015) Generating new test instances by evolving in instance space. *Computers & Operations Research* 63: 102–113.

References II

6. Smith-Miles K, Lopes L (2012) Measuring instance difficulty for combinatorial optimization problems. *Computers & Operations Research* 39: 875–889.

7. Smith-Miles K, Baatar D, Wreford B, et al. (2014) Towards objective measures of algorithm performance across instance space. *Computers & Operations Research* 45: 12–24.

8. Hogg T (2002) Adiabatic quantum computing for random satisfiability problems. *Phys Rev A 67 022314 (2003)*.

9. Young AP, Knysh S, Smelyanskiy VN (2009) First order phase transition in the quantum adiabatic algorithm. *Phys Rev Lett 104, 020502 (2010)*.

10. Farhi E, Goldstone J, Gosset D, et al. (2009) Quantum adiabatic algorithms, small gaps, and different paths. *arXiv preprint arXiv:09094766*.

References III

11. Latorre JI, Orús R (2004) Adiabatic quantum computation and quantum phase transitions. *Physical Review A* 69: 062302.

12. Hauke P, Bonnes L, Heyl M, et al. (2015) Probing entanglement in adiabatic quantum optimization with trapped ions. *Frontiers in Physics* 3: 21.

13. Lucas A (2014) Ising formulations of many np problems. *Frontiers in Physics* 2: 5.

14. Mandrà S, Katzgraber HG (2017) A deceptive step towards quantum speedup detection. *Quant Sci Technol 3, 04LT01 (2018)*.

15. Gabor T, Zielinski S, Feld S, et al. (2019) Assessing solution quality of 3SAT on a quantum annealing platform, In, *International workshop on quantum technology and optimization problems*, Springer, 23–35.

16. Li RY, Gujja S, Bajaj SR, et al. (2019) Unconventional machine learning of genome-wide human cancer data. *arXiv preprint arXiv:190906206*.

References IV

17. Inoue D, Okada A, Matsumori T, et al. (2020) Traffic signal optimization on a square lattice using the d-wave quantum annealer. *arXiv preprint arXiv:200307527*.