Quantum Bayesian Network Structure Learning

Lukas Rüttgers Mingxi Xie Pengfei Zhu Ziyi Xie Kairui Ding

Report by Pengfei Zhu Jan 7, 2024

Abstract

Bayesian network structure learning has wide applications in various fields from machine learning to biology. The goal is to learn the structure of a Bayesian network from data samples, which is NP-complete partly because the number of possible structures grows exponentially. Classical algorithms fall largely into two categories: constraint-based methods and score-based methods. For the former, we propose the quantum SGS algorithm, but it may not be feasible on NISQ devices. For the latter, we implemented a quantum annealing technique from a previous paper and also made some important improvements. We performed some experiments on both generated data and real-world data.

1 Problem formulation

We assume some familiarity with Bayesian networks. A brief introduction is provided in Appendix B.

We wish to learn the structure of a Bayesian network from data D sampled from the joint distribution $\Pr(\mathbf{X})$, up to a Markov equivalence class. It is not possible to discern only from data which structure in the equivalence class is the *true* structure, e.g. the causal structure, because all the structures will accommodate the same joint distributions. As always, one must exercise caution when inferring causality from correlation.

This problem has diverse real-world applications such as the short-term prediction of solar flares and the discovery of gene regulatory networks. Some other fields include machine learning (where Bayesian networks are commonly utilized for reasoning), air pollution modeling, etc.

We note that the problem is somewhat unique in that whether or not the data D fits a certain structure is not well-defined, and different types of algorithms tend to have different definitions. For example, a nearly fully connected network with few conditional independences may be able to accommodate a wide range of joint distributions, but may not be considered minimal for some distributions. Regardless, both score-based definitions [Chickering, 1996] and constraint-based definitions [Chickering et al., 2004] have been shown

to be NP-complete. This is partly because the search space is exponentially large, as the number of DAGs with n vertices is given by [Kotesovec, 2013]

$$a(n) \sim \frac{n!\ 2^{n(n-1)/2}}{Mp^n}, \quad M \approx 0.574,\ p \approx 1.488.$$

2 Classical algorithms

Classical algorithms for Bayesian network structure learning fall largely into three classes: constraint-based methods, score-based methods, and hybrid methods. We discuss the former two categories.

2.1 Constraint-based methods

With constraint-based methods, the idea is to use statistical conditional independence tests (CI tests) to find the conditional independence relations present in the data, and reconstruct the DAG based on these relations (constraints).

2.1.1 CI tests

For two nodes A, B and set S, for any combination of values a, b, s, the expected frequencies (if $A \perp B \mid S$) and observed frequencies are given as

$$\text{Expected}_{abs} = \frac{N_{bs}N_{as}}{N_s},\ \text{Observed}_{abs} = N_{abs}.$$

One simple CI test is the χ^2 test [Kitson et al., 2023]:

$$\chi^2 = 2 \cdot \sum_{a,b,s} \frac{(\text{Observed}_{abs} - \text{Expected}_{abs})^2}{\text{Expected}_{abs}}.$$

The metric is then compared to a threshold value, depending on the confidence level desired, to accept or reject the null hypothesis that $A \perp B \mid S$.

2.1.2 SGS algorithm

The SGS algorithm is rather inefficient but illustrates the key ideas of this class of algorithms. With some further assumptions (namely, faithfulness and causal sufficiency), the following two theorems can be derived from the definition of Bayesian networks [Verma and Pearl, 1990]:

- If $A \not\perp B \mid \mathbf{S}$ for every subset $\mathbf{S} \subseteq \mathbf{X} \backslash \{A, B\}$ in the DAG, then A and B are adjacent in the (undirected) graph.
- If A and B, and B and C are adjacent but A and C are not, then if $A \not\perp C \mid \mathbf{S} \cup B$ for any subset $\mathbf{S} \subseteq \mathbf{X} \backslash \{A, B, C\}$ in the DAG, then A, B, C form a *v-structure* $A \to B \leftarrow C$.

The SGS algorithm uses these theorems to learn the PDAG (Markov equivalence class). We start from a complete undirected graph.

- Adjacency phase: For each pair of nodes A, B, perform a CI test of A and B on every subset $\mathbf{S} \subseteq \mathbf{X} \backslash \{A, B\}$. If for any $\mathbf{S}$ the CI test says conditionally independent, remove the edge between A and B.
- V-structure phase: For every tuple A, B, C where A, B and B, C are adjacent but A, C are not, perform a CI test of A and C on $\mathbf{S} \cup B$ for every subset $\mathbf{S} \subseteq \mathbf{X} \backslash \{A, B, C\}$. If for every $\mathbf{S}$ the CI test says conditionally dependent, mark the *v-structure* $A \to B \leftarrow C$.

2.2 Score-based methods

Score-based methods use a score function as the objective function and reduce the problem to an optimization problem for the highest-scoring network structure.

From Bayes' Law, the likelihood of a network structure G given data D is

$$p(G \mid D) = \frac{p(D \mid G)p(G)}{p(D)}.$$

Recall that a Bayesian network consists of the network structure and network parameters Θ which encode the conditional probability tables. When deriving $p(D \mid G)$, it is necessary to make assumptions on the distribution of Θ. In *Bayesian scores*, Θ is assumed to follow a prior distribution, so overly complex models are implicitly penalized as the extra edges will likely not fit the prior very well. On the other hand, *information-theoretic scores* assume optimal selection of Θ and use an explicit penalty term on model complexity.

2.2.1 Bayesian scores

With Dirichlet priors for Θ and some further assumptions (namely multinomial sampling and parameter independence and modularity), we can derive the Bayesian Dirichlet (BD) score [Heckerman et al., 1995]:

$$S_{\mathrm{BD}}(G, D) = \log p(G \mid D) = \log P(G) + \sum_{i=1}^{n} \sum_{j=1}^{q_i} \left[\log \frac{\Gamma(N_{ij})}{\Gamma(N_{ij} + N'_{ij})} + \sum_{k=1}^{r_i} \log \frac{\Gamma(N_{ijk} + N'_{ijk})}{\Gamma(N'_{ijk})} \right]$$

where r_i is the number of possible values for X_i (2 in the binary case), and q_i is the number of possible combinations of values for X_i's parents (in the binary case, $2^{|\mathbf{S}|}$ where $\mathbf{S}$ is the parent set). Here j iterates over all q_i possible combinations for X_i's parents, and N_{ij} is the number of instances in D where X_i's parents takes the j-th possible combination of values; and N_{ijk} is the number of instances out of N_{ij} where X_i takes the k-th possible value. N' is the prior belief on the data. Setting $N'_{ijk} = 1$ gives the simplified version, the K2 score:

$$S_{\mathrm{K2}}(G, D) = \log P(G) + \sum_{i=1}^{n} \sum_{j=1}^{q_i} \left[\log \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} + \sum_{k=1}^{r_i} \log(N_{ijk}!) \right].$$

2.2.2 Information-theoretic scores

The simplest class, known as the *Bayesian information criterion* scores, have the general form [Kitson et al., 2023]

$$S(G, D) = \log[\hat{p}(D \mid G)] - \Delta(D, G)$$

where $\hat{p}(D \mid G)$ is the Bayesian likelihood assuming the *maximum likelihood estimation* of network parameters Θ, and $\Delta(D, G)$ is the penalty for model complexity.

3 Quantum SGS algorithm

During our discussions, a prototypical idea was proposed for a quantum algorithm based on the SGS algorithm. In both the adjacency phase and the *v-structure* phase, CI tests are performed on all subsets $\mathbf{S}$ to check if there's an edge or *v-structure*. The idea is to parallelize these CI tests: If we can represent the CI test as a unitary, we will only need to check if the unitary is constant. This may allow us to achieve some speedup by applying the unitary to a superposition of subset specifiers, as in Grover's algorithm.

3.1 CI test as a unitary

Suppose that we start with an absolute frequency table $D_{000\cdots000}$ to $D_{111\cdots111}$, where the indices are binary strings of length n and each entry is the number of occurrences of that combination (that binary string) in the data. To calculate N_{abs}, we need to *marginalize out* variables not in $\mathbf{S} \cup \{A, B\}$. To do that, for each variable $i \notin \mathbf{S} \cup \{A, B\}$, we add $D_{x_0\cdots x_{i-1}1x_{i+1}\cdots x_{n-1}}$ to $D_{x_0\cdots x_{i-1}0x_{i+1}\cdots x_{n-1}}$ for all $x_0 \cdots x_{i-1}x_{i+1} \cdots x_{n-1}$. This step may be implemented as $(n-2)2^{n-1}$ additions controlled by $\mathbf{S}$ specifier qubits. Finally, the result of N_{abs} can be accessed by setting the values for variables not in $\mathbf{S} \cup \{A, B\}$ to 0. (For example, if A is X_0, B is X_1, and $\mathbf{S} = \{X_2, \cdots, X_{2+m}\}$, N_{abs} will be in $D_{abs_0\cdots s_{m-1}000\cdots000}$.)

Now that we have N_{abs}, each term in χ^2 may be calculated *locally* for each choice of a, b, s. This is done for the whole D (2^{n-2} gates), even though some values (those that have been marginalized out) will not be used. Then we need to sum up the useful values, i.e. marginalize out variables *in* $\mathbf{S} \cup \{A, B\}$. This step can be implemented similarly to before, or alternatively as 2^n controlled additions to an auxiliary accumulator: $D_{x_0\cdots x_{n-1}}$ is added to the accumulator if for all $x_i = 1$, $X_i \in \mathbf{S} \cup \{A, B\}$. Finally, the accumulator result is compared with a threshold.

An example circuit with $n = 3$ is shown in Appendix C.

3.2 Checking if the unitary is constant

This problem is known as the quantum existence problem and is a special case of the quantum counting problem. With Grover's algorithm and quantum phase estimation, it is possible to solve this problem in $O(\sqrt{N})$ oracle calls [Nielsen and Chuang, 2001] compared to $\Omega(N)$ required on a classical computer.

3.3 Total complexity

The total complexity for the adjacency phase or v-structure phase on a single pair of nodes A, B would be $O(\sqrt{2^n} \cdot n2^n) = O(n2^{3n/2}) = O(n\left(2^{3/2}\right)^n) \approx O(n \cdot 2.828^n)$.

In comparison, a naïve classical implementation to compute N_{abs} will require summing up $2^{n-2-|\mathbf{S}|}$ entries. This needs to be done for each combination of a, b, s, for $2^{2+|\mathbf{S}|}$ combinations. Hence a single CI test of nodes A and B on $\mathbf{S}$ will take 2^n additions. To perform the tests on all $\mathbf{S}$ would be 2^{2n} additions. There may be some clever indexing techniques to improve the classical implementation (but the one on the slides would be incorrect).

In any case, we believe that our quantum SGS algorithm may have an advantage compared to the classical SGS algorithm. Of course, the SGS algorithm is the most basic algorithm and is not very efficient even with a quantum advantage. However, our techniques may be applied to more advanced algorithms as well, so long as they need to perform CI tests on several subsets.

One clear downside is that the number of qubits required is quite large, and the circuit will likely be very complicated and sensitive to noise, as integer and decimal addition and multiplication are required. Therefore, we believe that this idea is likely not feasible for NISQ devices and may be more suited to error-corrected large-scale quantum computers.

4 Quantum annealing approach

Our work on quantum annealing builds upon the approach presented in [O'Gorman et al., 2015]. In that paper the Hamiltonian is made up of four parts:

$$H(d, y, r) = H_{\text{score}}(d) + H_{\max}(d, y) + H_{\text{trans}}(r) + H_{\text{consist}}(d, r)$$

where the qubits used are

$$|\{d_{ij}\}| = n(n-1),\ |\{r_{ij}\}| = \frac{n(n-1)}{2},\ |\{y_{il}\}| = n\mu = n\left\lfloor \log_2(m+1) \right\rfloor.$$

The d qubits represent the adjacency matrix; the r qubits represent a topological order and is used to ensure that the graph is a DAG; and the y qubits are used to limit the maximum in-degree of nodes.

We modified H_{score} to reduce its locality. We also removed the y qubits, replacing $H_{\max}(d, y)$ with a *norm penalty* $H_{\text{norm}}(d)$.

4.1 Score Hamiltonian

For each node X_i, the sub-scores for each possible choice of parent set J can be precomputed classically. Let this be denoted $s_i(J)$. The function of choice here is the K2 score.

Let the actual parent set of X_i as specified in d_{ij} be $J_i^*(d)$, Then the score Hamiltonian can be expressed as

$$H_{\text{score}}^{(i)} = s_i(J_i^*(d)) = \sum_{J \subset V \setminus \{i\}} s_i(J) \mathbb{1}[J = J^*] = \sum_{J \subset V \setminus \{i\}} \left[s_i(J) \prod_{j \in J} d_{ji} \prod_{j \notin J, j \neq i} (1 - d_{ji}) \right].$$

This naïve expression is n-local. If we set a maximum in-degree $|J| \leq m$, [O'Gorman et al., 2015] was able to achieve m-locality with a trick inspired by the inclusion-exclusion principle:

$$w_i(J) = \sum_{l=0}^{|J|} (-1)^{|J|-l} \sum_{K \subset J, |K|=l} s_i(K), \quad H_{\text{score}}^{(i)} = \sum_{J \subset V \setminus \{i\}, |J| \leq m} \left[w_i(J) \prod_{j \in J} d_{ji} \right].$$

w_i is added not only for J^*, but for all sets $J \subseteq J^*$, but the sum evaluates to $s_i(J^*)$.

However, if $m > 2$, $O(n^m)$ auxiliary qubits will be needed to convert to a 2-local Ising model form. We proposed an alternative k-local Hamiltonian, approximating $\mathbb{1}[J = J^*]$ with a Hamming distance-based weight. Formally,

$$d(J, J^*) \equiv (n-1) - \sum_{j \in J} d_{ji} + \sum_{j \notin J, j \neq i} (1 - d_{ji}), \quad H_{\text{score}}^{(i)} = \sum_{J \subset V \setminus \{i\}} s_i(J) \left[\frac{n - d(J, J^*)}{n} \right]^k.$$

Here we raise the weight to the k-th power so that it approaches 0 quicker and approximates $\mathbb{1}[J = J^*]$ better. In Appendix D, we present some experimental results to demonstrate that the approximation is acceptable, and that $k = 2$ is more effective than $k = 1$.

4.2 Penalty terms

To ensure that the graph is a DAG, boolean variables r_{ij} are introduced to represent a topological order of the graph. Every DAG admits at least one topological ordering, while a cyclic graph admits none. Two types of penalty terms are employed: [O'Gorman et al., 2015]

- *Transitive Hamiltonian* to ensure that r is an ordering and there is no cycle. Namely,

  $$H_{\text{trans}}^{(ijk)}(r_{ij}, r_{ik}, r_{jk}) \equiv \delta_{\text{trans}}^{(ijk)} \left[r_{ij} r_{jk} (1 - r_{ik}) + (1 - r_{ij})(1 - r_{jk}) r_{ik} \right]$$

  such that the term is 0 if r is transitive among i, j, k and positive otherwise.

- *Consistency Hamiltonian* to ensure that r is consistent with the graph d. Namely,

  $$H_{\text{consist}}^{(ij)}(d_{ij}, d_{ji}, r_{ij}) = \delta_{\text{consist}}^{(ij)} [d_{ji} r_{ij} + d_{ij}(1 - r_{ij})]$$

  such that the term is 0 if r_{ij} is consistent with d_{ij} and d_{ji}, and positive otherwise.

In [O'Gorman et al., 2015], a *max Hamiltonian* is introduced to limit the in-degree of nodes to m such that it is consistent with the m-local score Hamiltonian and to prevent the graph from being too dense. However, $n\mu$ additional qubits are required for this, which will significantly hinder our simulation. We also observed that despite the score function (K2) being a Bayesian score (which should implicitly penalize model complexity), most of the time the resulting graph still had superfluous edges, and a hard limit on in-degree did not help much. Therefore, we propose to remove these qubits and replace the max Hamiltonian with a *norm Hamiltonian*, defined as

$$H_{\text{norm}}(d) = \lambda \|d\|_k, \quad \|d\|_k = \left[\sum_{i,j} |d_{ij}|\right]^k$$

where k is the desired locality. Appendix E shows some experiments with the new term.

The obvious downside is one extra parameter to tune. We also considered switching to an information-theoretic score with explicit penalty terms on model complexity. However, those terms are typically linear in the number of free parameters and exponential in the number of edges, and will not be easily realizable as a 2-local Hamiltonian. Therefore, we ultimately decided to use a Bayesian score to take advantage of its implicit (albeit not strong enough) penalty and add ad-hoc norm penalties where necessary.

4.3 Lower bounds of penalty weights

The penalty weights δ need to be set appropriately such that an illegal state should never be the ground state. In [O'Gorman et al., 2015], the authors have given the bounds

$$\begin{aligned}
\delta^{(i)}_{\text{max}} &> \max_{j \neq i} \Delta_{ji}, 1 \leq i \leq n \\
\delta^{(ij)}_{\text{consist}} &> (n-2) \max_{k \neq (i,j)} \delta^{(ijk)}_{\text{trans}}, 1 \leq i < j \leq n \\
\delta^{(ijk)}_{\text{trans}} = \delta_{\text{trans}} &> \max_{\substack{1 \leq i',j' \leq n \\ i' \neq j'}} \Delta_{i'j'}, 1 \leq i < j < k \leq n
\end{aligned}$$

where $\Delta_{ji} = \max\{0, \Delta'_{ji}\}$, and

$$\Delta'_{ji} = \max_{\{d_{ki} | k \neq i,j\}} \{H^{(i)}_{score}|_{d_{ji}=0} - H^{(i)}_{score}|_{d_{ji}=1}\}$$

is the largest reduction in the score function for adding an edge ji.

The authors also provided a bound for Δ'_{ji} for the m-local score Hamiltonian. For our Hamming distance approximation, we derive the new bounds in Appendix F.

In practice, setting δ to a value close to the lower bounds will still often yield illegal results. Therefore we set δ to $\max\{F \cdot \delta_{\min}, B\}$, where F is a factor (1.5 to 2 seems to be good) and B is a manually-specified hard lower bound. It was still necessary to tune B a bit for our experiments.

5 Experiments

5.1 Toy experiments

These experiments are conducted on randomly generated data according to the ground truth network structure. Quadratic Hamming distance approximation is used. We recorded whether or not the top QA result is legal (i.e. is a DAG and r is consistent with d), whether or not it matches the ground truth structure (up to a Markov equivalence class), and the count of the top result in the measurement results out of 1000 shots.

For $n = 4$, structure 2 is a chain $A \to B \to C \to D$. We believe this may be harder to learn as the causal effects are dampened. Notably, at $\lambda = 1$ the top QA result is a partial chain $A \to B \to C$.

	λ	Legal?	Correct?	Frequency
$n = 2$	0	Yes	Yes	498/1000
$n = 3$	0	Yes	Yes	110/1000
$n = 4$, structure 1	5	Yes	Yes	19/1000
$n = 4$, structure 2	0	Yes	No	21/1000
$n = 4$, structure 2	1	Yes	No	49/1000

5.2 Real-world data

We used numerical and binary medical data of ICU patients in US hospitals from `https://www.kaggle.com/c/widsdatathon2020/data`. Numerical data are transformed into binary ones by choosing an appropriate threshold.

(a) diabetes mellitus, immunosuppression and cirrhosis

(b) Age ($\geq$ 60), BMI ($\geq$ 28), diabetes and hepatic failure; $\lambda = 20$

Figure 1: Analysis of real-world data

References

[Chickering, 1996] Chickering, D. M. (1996). *Learning Bayesian Networks is NP-Complete*, pages 121–130. Springer New York, New York, NY.

[Chickering et al., 2004] Chickering, M., Heckerman, D., and Meek, C. (2004). Large-sample learning of bayesian networks is np-hard. *Journal of Machine Learning Research*, 5:1287–1330.

[Heckerman et al., 1995] Heckerman, D., Geiger, D., and Chickering, D. M. (1995). Learning bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20:197–243.

[Kitson et al., 2023] Kitson, N. K., Constantinou, A. C., Guo, Z., Liu, Y., and Chobtham, K. (2023). A survey of bayesian network structure learning. *Artificial Intelligence Review*, pages 1–94.

[Kotesovec, 2013] Kotesovec, V. (2013). Number of acyclic digraphs (or DAGs) with n labeled nodes: Formula. `https://oeis.org/A003024`.

[Nielsen and Chuang, 2001] Nielsen, M. A. and Chuang, I. L. (2001). *Quantum computation and quantum information*, chapter 6, page 263. Cambridge University Press.

[O'Gorman et al., 2015] O'Gorman, B., Babbush, R., Perdomo-Ortiz, A., Aspuru-Guzik, A., and Smelyanskiy, V. (2015). Bayesian network structure learning using quantum annealing. *The European Physical Journal Special Topics*, 224:163–188.

[Verma and Pearl, 1990] Verma, T. and Pearl, J. (1990). Causal networks: Semantics and expressiveness. In *Machine intelligence and pattern recognition*, volume 9, pages 69–76. Elsevier.

A My contributions to the project

I participated in class discussions and prepared portions of many of the slides. I reviewed Ziyi Xie's original code implementation and investigated and fixed multiple bugs to make it work. (For this I coded a brute-force solver.) I implemented Lukas's Hamming distance approximations and designed and carried out experiments on the landscape of the Hamiltonians. I proposed to replace the y qubits with a novel penalty term and performed experiments to support the idea. In the final presentation, I was responsible for the toy experiments and also polished up the other parts.

B Preliminaries

Suppose that we have random variables $\mathbf{X} = (X_1, \cdots, X_n)$. The random variables may be discrete or continuous. In this project, we only consider discrete random variables, and in particular, we mainly consider binary random variables.

B.1 Bayesian networks

A Bayesian network B is a tuple (G, Θ) of a directed acyclic graph G and conditional probability tables Θ. The vertices in G correspond to random variables X_i. While Bayesian networks are often construed to represent causal relationships, in the plain model, the graph simply represents conditional dependence. An edge $A \to B$ indicates a direct conditional dependence from A to B. We say that A is a *parent* of B. The probability tables Θ specify the distributions for each node X_i given any combination of values for its parents.

What Bayesian networks truly encode are conditional dependence and independence. In particular, two assumptions are made: [Kitson et al., 2023]

- Markov Condition: Every variable X in G is conditionally independent of any non-descendant, given its parents.
- Minimality: No edges in the DAG can be removed without implying a new conditional independence relationship not present in $\Pr(\mathbf{X})$. In other words, no edges can be removed such that the DAG can still accommodate $\Pr(\mathbf{X})$.

B.2 Markov equivalence classes

For a given Bayesian network structure, certain further conditional independences are implied in addition to the direct assumptions of the Markov Condition. For three nodes, the three types of causal structures are shown in Figure 2. Structures such as (c) are called *v-structures*.

We observe that certain Bayesian network structures (DAGs) encode the same conditional independence relations, such as (a) (b) in Figure 2. We say that they belong to the

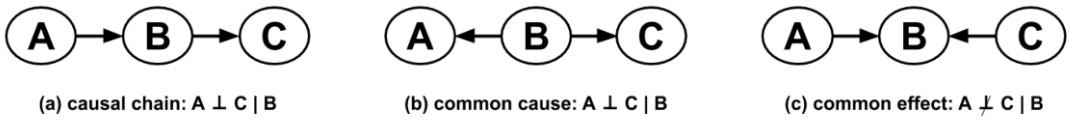


Figure 2: Causal structures with three nodes [Kitson et al., 2023]

same *Markov equivalence class*. [Verma and Pearl, 1990] showed that the *skeleton* (undirected graph) and the *v-structures* completely determine the conditional independences of a Bayesian network.

In Figure 3 (a), all three network structures are equivalent. We may use a Partially Directed Acyclic Graph (PDAG) to represent this equivalence class, where only the edges in a *v-structure* are directed, as in Figure 3 (b).

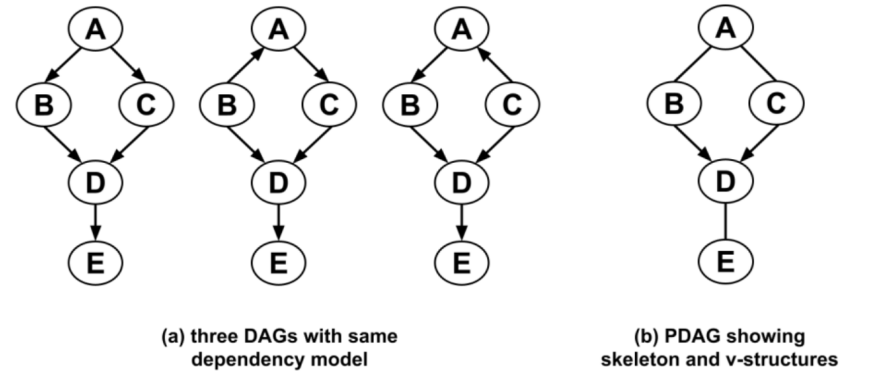


Figure 3: Markov equivalence classes [Kitson et al., 2023]

C Example circuit for CI test as a unitary

See Figure 4. We assume each integer or real number is expressed as k qubits. Wires that pass through a gate are not inputs to the gate and are not affected by the gate.

Here the '+' gate adds the lower number to the upper. The 'χ^2' gate takes as input N_{00s}, N_{01s}, N_{10s}, N_{11s} and gives output T_{00s}, T_{01s}, T_{10s}, T_{11s} in the corresponding wires, where

$$T_{abs} = 2\frac{(\text{Observed}_{abs} - \text{Expected}_{abs})^2}{\text{Expected}_{abs}}, \quad \text{Expected}_{abs} = \frac{N_{bs}N_{as}}{N_s}, \quad \text{Observed}_{abs} = N_{abs}.$$

The '$< T$' gate compares the χ^2 value with the threshold T.

We do not show the implementations of these gates as they will be very complicated, but it should be clear that they can be realized with poly(k) basic gates.

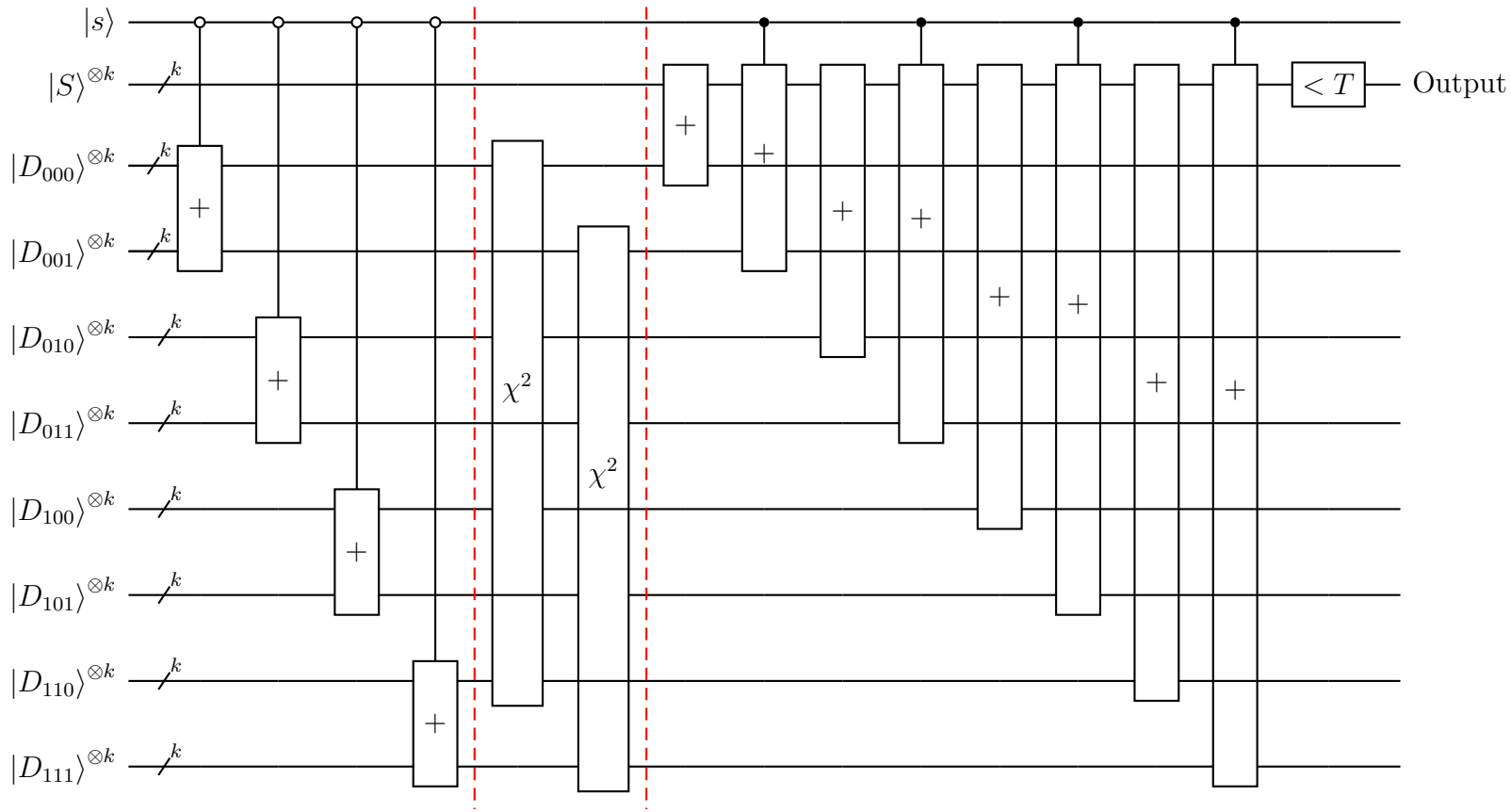


Figure 4: CI test as a unitary, $n = 3$ example

D Experiments with the Hamming distance approximation

We note several potential problems with the approximation:

- It may be argued that there are exponentially many possibilities as the distance increases, but the weight only decreases polynomially, and the total 'influence' on the score increases as the distance increases. However, this is not a problem as all the subscores are precalculated and we are just trying to find a 'center' for the weight in the state space such that the total weighted score is minimized.
- The score of a state will now depend on the scores of states in its neighborhood, and hence the global minimizer is not necessarily preserved.
- The approximation may introduce new local minima and cause difficulty in optimization. Alternatively, it may smoothen the landscape and make optimization easier.

D.1 Minimizer

Shown in Figure 5 are brute force ground states with different Hamiltonians. Since $\lambda = 0$, there exist many states with similar energies. We note that neither approximation preserved the minimizers, but the linear approximation changed the minimizer to a different equivalence class, whereas the quadratic approximation changed the minimizer to a different graph in the same equivalence class. Both cases may be acceptable since the energy levels are very similar: with the original Hamiltonian, the minimizer in (b) has an energy of 254.72; the minimizer in (c) has energy 256.62; the minimizer in (d) has energy 255.73. We postulate that setting a larger λ may help ensure that all minimizers are in the same equivalence class since extra edges will incur a penalty.

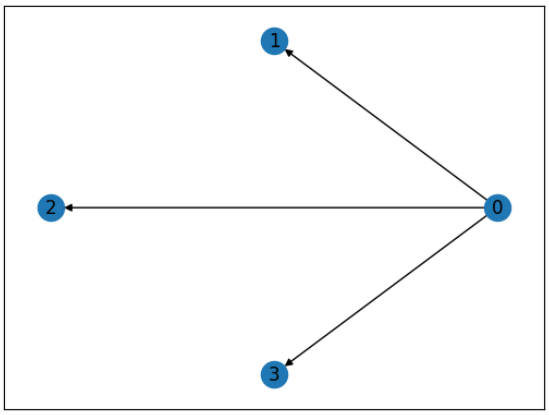
D.2 Hamiltonian landscape

The state space has a very high dimension. To visualize the landscape of the different Hamiltonians, we categorize the states by two different distance metrics:

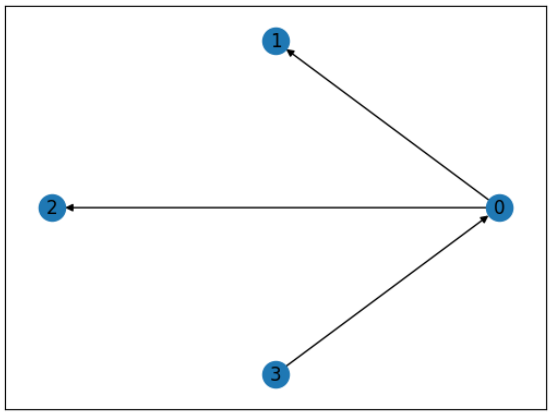
- Spin distance: the number of bits flipped.
- DAG distance: the number of bits flipped in d. This only counts legal states, i.e. d is a DAG and r is consistent with d.

We then plotted the minimum and mean energy gaps (from the ground state) of the states in each category (distance).

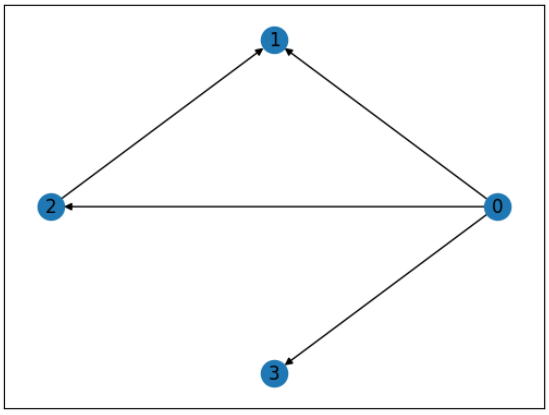
Figure 6 shows the minimum energy gap by spin distance. The plot for $k = 2$ is very similar to the original Hamiltonian, while the $k = 1$ plot is slightly different. This may be because of the different minimizers.



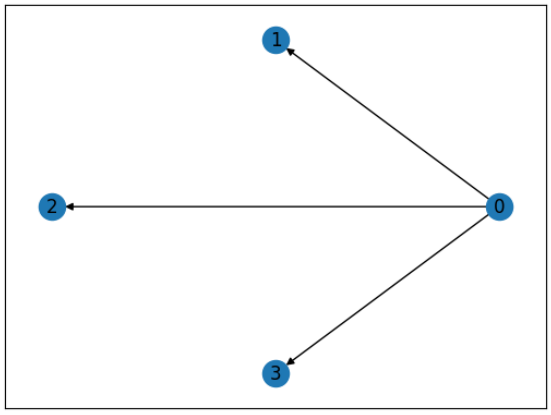
(a) Ground truth network structure



(b) Original Hamiltonian

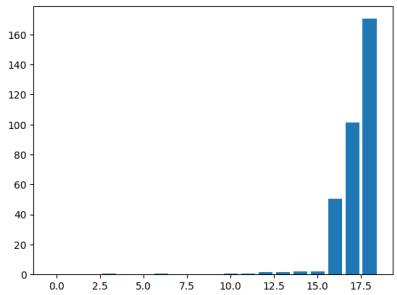


(c) $k = 1$

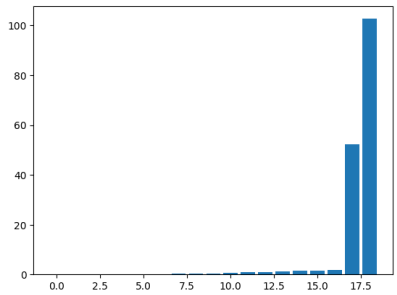


(d) $k = 2$

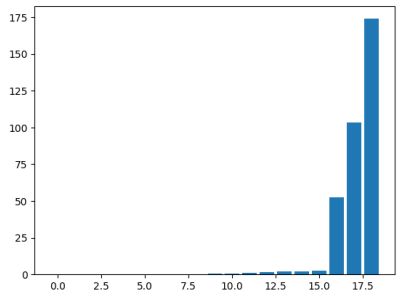
Figure 5: Minimizer experiments



(a) Original Hamiltonian



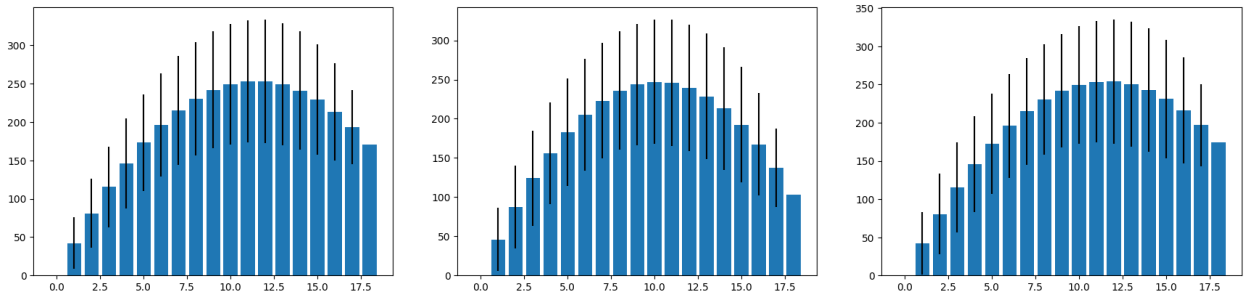
(b) $k = 1$



(c) $k = 2$

Figure 6: Minimum energy gap by spin distance

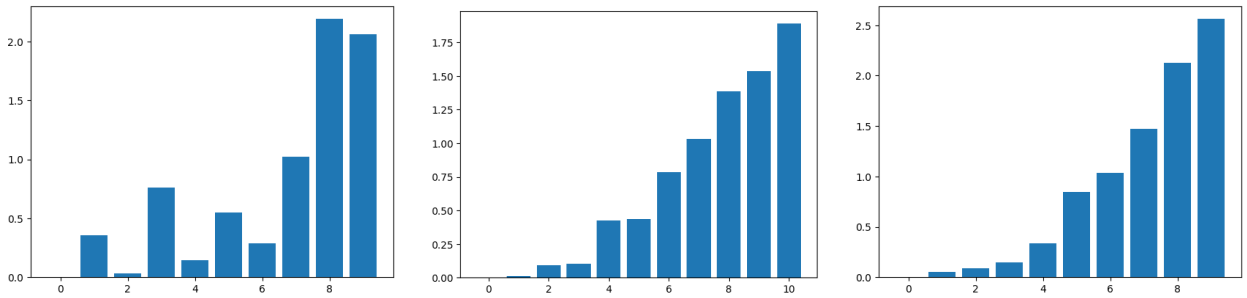
Figure 7 shows the mean energy gap by spin distance and the variance. The $k = 2$ case is again very similar to the original Hamiltonian, though we do notice that the variance is slightly larger.



(a) Original Hamiltonian (b) $k = 1$ (c) $k = 2$

Figure 7: Mean energy gap by spin distance

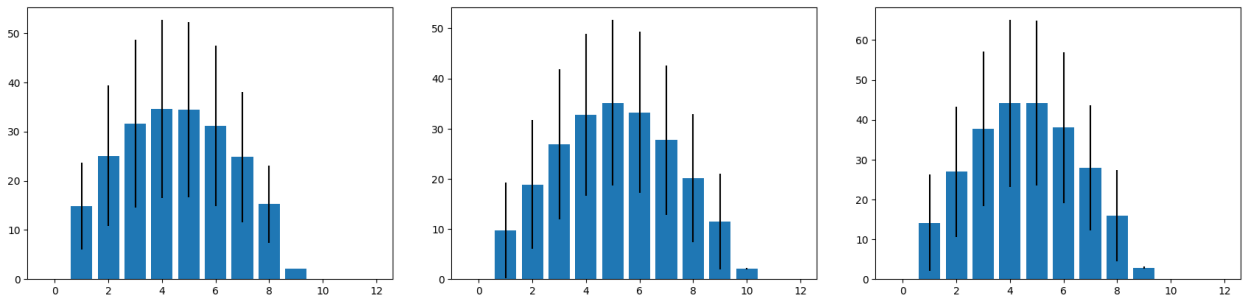
Figure 8 shows the minimum energy gap by DAG distance. Here we can see some smoothening effect, where the landscape appears more smooth in the approximations. The effect seems to be more obvious at $k = 2$.



(a) Original Hamiltonian (b) $k = 1$ (c) $k = 2$

Figure 8: Minimum energy gap by DAG distance

Figure 9 shows the mean energy gap by DAG distance. The three plots seem similar, though the variance is larger in the approximations.

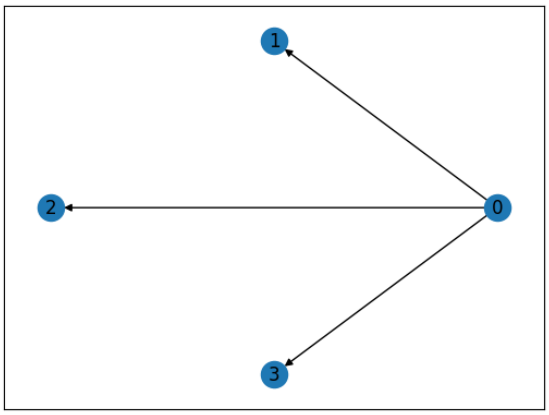


(a) Original Hamiltonian (b) $k = 1$ (c) $k = 2$

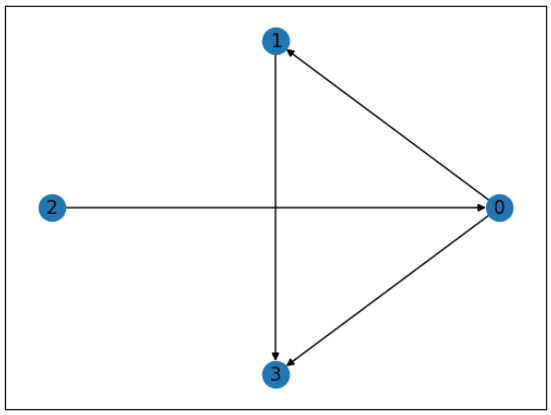
Figure 9: Mean energy gap by DAG distance

E Experiments with the norm Hamiltonian

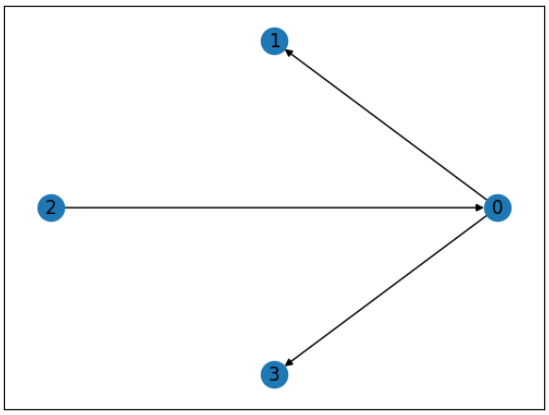
Shown in Figure 10 are brute force ground states, with the original score Hamiltonian (without the Hamming distance approximation). We can see that the introduction of the norm penalty allows us to tune it to retain the most important edges.



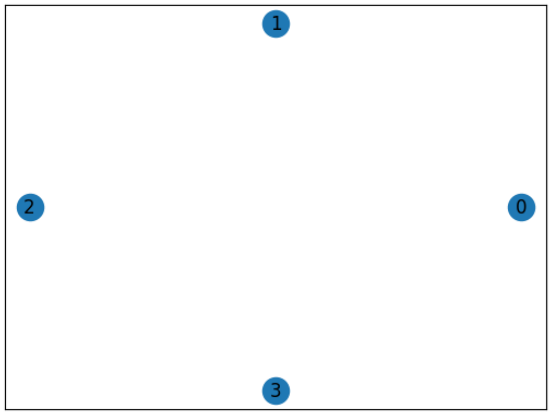
(a) Ground truth network structure



(b) $m = 2$, $\lambda = 0$



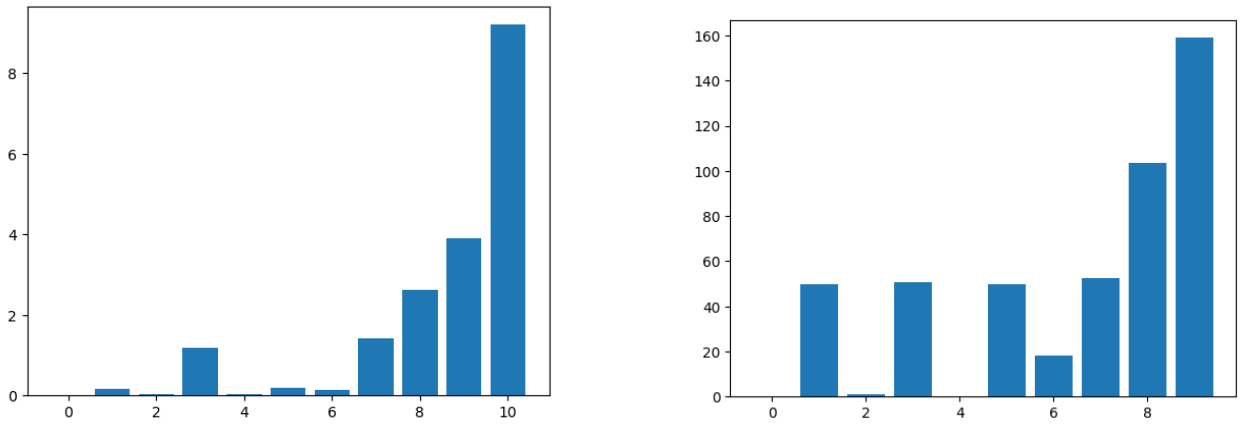
(c) $m = 2$, $\lambda = 50$



(d) $m = 2$, $\lambda = 500$

Figure 10: Norm Hamiltonian experiments

In Figure 11 we show the total Hamiltonian landscape by DAG distance. The local minima seem 'deeper' with the new penalty term. This is not necessarily a bad thing since other local minima are also quite good. With the new term, the optimization may more easily tend to one of the good minima with no superfluous edges.



(a) $\lambda = 0$ (b) $\lambda = 50$

Figure 11: Total Hamiltonian landscape with the norm penalty

F Lower bounds of penalty weights for the new score Hamiltonian

Recall that

$$H^{(i)}_{score}(d_i) \equiv \sum_{J \subset V \setminus \{i\}} \left(\frac{n - d(J, J^*)}{n} \right)^k \cdot s_i(J).$$

Then

$$\begin{aligned}
&H^{(i)}_{score}|_{d_{ji}=0} - H^{(i)}_{score}|_{d_{ji}=1} \\
&= \sum_{J \subseteq V \setminus \{i,j\}} \left\{ \left(\frac{n - d(J, J^*)}{n} \right)^k \Big|_{d_{ji}=0} - \left(\frac{n - d(J, J^*)}{n} \right)^k \Big|_{d_{ji}=1} \right\} \cdot s_i(J) \\
&+ \sum_{J \subseteq V \setminus \{i,j\}} \left\{ \left(\frac{n - d(J \cup \{j\}, J^*)}{n} \right)^k \Big|_{d_{ji}=0} - \left(\frac{n - d(J \cup \{j\}, J^*)}{n} \right)^k \Big|_{d_{ji}=1} \right\} \cdot s_i(J \cup \{j\})
\end{aligned}$$

Let

$$\alpha(j, J, J^*) := \frac{1}{n} \left(\sum_{k \in J \setminus \{j\}} d_{ki} + \sum_{k \notin J \setminus \{i,j\}} (1 - d_{ki}) \right)$$

then $\alpha(j,J,J^*) \in \left[0, \frac{n-2}{n}\right)$ and

$$\begin{aligned}
&H^{(i)}_{score}|_{d_{ji}=1} - H^{(i)}_{score}|_{d_{ji}=0} \\
&= \sum_{J \subseteq V \setminus \{i,j\}} \left\{ \left(\alpha(j,J,J^*) + \frac{1}{n} \right)^k - (\alpha(j,J,J^*))^k \right\} \cdot s_i(J) \\
&+ \sum_{J \subseteq V \setminus \{i,j\}} \left\{ (\alpha(j,J,J^*))^k - \left(\alpha(j,J,J^*) + \frac{1}{n} \right)^k \right\} \cdot s_i(J \cup \{j\}).
\end{aligned}$$

Since $k \geq 1$, we have $(\alpha(j,J,J^*))^k - \left(\alpha(j,J,J^*) + \frac{1}{n}\right)^k \leq \frac{k}{n}$ and

$$\begin{aligned}
\Delta'_{ji} &= \max_{\{d_{ki} | k \neq i,j\}} \{ H^{(i)}_{score}|_{d_{ji}=0} - H^{(i)}_{score}|_{d_{ji}=1} \} \\
&\leq -\frac{k}{n} \min \left\{ \sum_{J \subseteq V \setminus \{i,j\}} s_i(J \cup \{j\}), 0 \right\} + \frac{k}{n} \max \left\{ \sum_{J \subseteq V \setminus \{i,j\}} s_i(J \cup \{j\}), 0 \right\}.
\end{aligned}$$