Quantum Speedups for Bayesian Network Structure Learning

Juha Harviainen[1] **Kseniya Rychkova**[2] **Mikko Koivisto**[1]

[1]University of Helsinki
[2]University of Queensland

Abstract

The Bayesian network structure learning (BNSL) problem asks for a directed acyclic graph that maximizes a given score function. For networks with n nodes, the fastest known algorithms run in time $O(2^n n^2)$ in the worst case, with no improvement in the asymptotic bound for two decades. Inspired by recent advances in quantum computing, we ask whether BNSL admits a polynomial quantum speedup, that is, whether the problem can be solved by a quantum algorithm in time $O(c^n)$ for some constant c less than 2. We answer the question in the affirmative by giving two algorithms achieving $c \leq 1.817$ and $c \leq 1.982$ assuming the number of potential parent sets is, respectively, subexponential and $O(1.453^n)$. Both algorithms assume the availability of a quantum random access memory. We also prove that one presumably cannot lower the base 2 for any classical algorithm, as that would refute the strong exponential time hypothesis.

1 INTRODUCTION

In the score-and-search approach to structure learning in Bayesian networks, one specifies a score function to be maximized over all possible directed acyclic graphs (DAGs) on a given node set. Common score functions—such as BDeu, BGe, BIC, fNML, or qNML—are decomposable: the score of a DAG is obtained by summing up the local scores of each node. The local score expresses how well the given parent set for a node fits the observed data, prior knowledge or constraints, and the adopted measures of learning success. See the textbook of Koller and Friedman [2009] for other approaches and the survey of Kitson et al. [2023, Sec. 4.1] for descriptions of the scores.

Decomposability motivates studying a more abstract problem formulation, in which the local scores are treated as the input, effectively ignoring that they originate from a particular scoring metric and observed data. This optimization problem, known as *Bayesian network structure learning* (BNSL), can be solved by dynamic programming over node subsets in time $O(2^n n^2)$, thus nearly linearly in the input size [Ott et al., 2004, Singh and Moore, 2005, Silander and Myllymäki, 2006]. But what if the input consists of significantly fewer local scores, e.g., each node can have at most some constant number of parents—a case relevant in practice? Unfortunately, essentially no faster algorithms are known, the base of the exponential bound being stuck at 2. In fact, the problem is NP-hard already if the maximum in-degree of the DAG is set to 2 [Chickering, 1995]. That said, there have been significant advances in heuristic algorithms, which may run fast for many practical instances [Yuan and Malone, 2013, Bartlett and Cussens, 2017], as well as in parameterized algorithms, which admit improved worst-case time bounds for restricted problem variants (see Grüttemeier and Komusiewicz [2022] and references therein).

Here, we ask whether quantum algorithms can beat the known exponential-time classical algorithms for BNSL. Quantum algorithms differ from classical ones in that they can harness quantum effects, such as superposition and entanglement. Typically (but not always) a quantum speedup is obtained by representing the problem in an appropriate way and then invoking a routine known as *quantum search* or *Grover's algorithm* [Grover, 1996]. Given a black-box mapping $f : \{1, 2, \ldots, m\} \to \{0, 1\}$, this routine only requires $O(\sqrt{m/k})$ evaluations of f to find, with high probability, an element that maps to 1, supposing there are k such elements; the expected number of evaluations required by any classical algorithm is linear in m/k. Several problems are known to admit a quadratic quantum speedup in relation to the best known classical algorithms, examples ranging from the satisfiability problem [Dantsin et al., 2005] to learning linear classifiers [Kapoor et al., 2016, Roget et al., 2022] and to reinforcement learning [Dunjko et al., 2016].

While there exist quantum *approximation* algorithms for BNSL [O'Gorman et al., 2015, Soloviev et al., 2023], ap-

parently, for *exact* BNSL no quantum speedup was known before the present work. The main challenge is that the best classical algorithms already are significantly faster than exhaustive search over the super-exponentially many DAGs. The dynamic programming algorithms resemble the Bellman–Held–Karp algorithm [Held and Karp, 1961, Bellman, 1962] for the traveling salesman problem and related "permutation problems" [Koivisto and Parviainen, 2010, Bodlaender et al., 2012], for which a quantum speedup was discovered only relatively recently [Ambainis et al., 2019].

Inspired by the results of Ambainis et al., we will show that BNSL admits a quantum algorithm running in time $O(1.817^n F)$, where $F \leq n2^{n-1}$ is the number of local scores given as input. This gives a polynomial speedup as long as F grows subexponentially or very moderately exponentially in n. To give a polynomial speedup also when F grows more rapidly, we present another, rather different algorithm: we make use of a construction previously given for trading space for time in a broad class of permutation problems [Koivisto and Parviainen, 2010], including BNSL [Parviainen and Koivisto, 2013]. We give a quantum algorithm running in time $O(1.982^n)$, provided that $F = O(1.453^n)$. Both algorithms require a *quantum random access memory* (QRAM) [Giovannetti et al., 2008], of which experimental implementations do not yet exist.

Could the base 2 be lowered also for a classical algorithm? Before the present work, the only evidence against has been the lack of progress in faster algorithms. Curiously, for a problem variant that ask for a sum over DAGs, the base of the exponential time bound was recently lowered from 3 [Tian and He, 2009] to 2.985 [Koivisto and Röyskö, 2020]. We will show that for the maximization variant, similar improvement presumably is not possible: we prove that it would refute the *strong exponential time hypothesis* (SETH) and thus give a way to solve the CNF-SAT problem on n variables in time $O(c^n)$ with some constant $c < 2$.

The rest of this paper is organized as follows. Section 2 introduces more formally the setup, namely the BNSL problem, the quantum search routine, and QRAM. Section 3 gives our first algorithm and Section 4 the second. The connection to SETH is presented in Section 5. In Section 6 we discuss some open problems and the role of QRAM.

2 PRELIMINARIES

This section introduces the main ingredients needed in later parts of the paper.

2.1 GRAPHS AND ORDERS

Let N be a finite set and $R \subseteq N \times N$. We let

$$R_i := \{j : ji \in R\}$$

denote the *parent set* (i.e., direct predecessors) of i in R.

If R is *acyclic*, i.e., there are no elements $i_1, i_2, \ldots, i_k$ such that $i_1 = i_k$ and $i_t i_{t+1} \in R$ for all $t = 1, 2, \ldots, k-1$, then (N, R) is a *directed acyclic graph* (DAG).

We call R a *partial order* on N if it is irreflexive and transitive, and a *linear order* on N if it is, in addition, total (aka strongly connected). A linear order L on N is a *linear extension* of R if $L \supseteq R$, i.e., $L_i \supseteq R_i$ for all $i \in N$.

These concepts are illustrated in Figure 1.

2.2 THE BNSL PROBLEM

Given a *node set* N of size n and a *local score* $s_i(J)$ for each node $i \in N$ and node subset $J \subseteq N \setminus \{i\}$, the BNSL problem is to find a DAG (N, A) that maximizes the score

$$s(A) := \sum_{i \in N} s_i(A_i)\,,$$

which measures how well the DAG fits the prior assumptions and the data. Here we identify the DAG with its *arc set* A, the node set N being fixed. Recall that A_i denotes the parent set of i. Since our algorithms work for any decomposable score, we do not specify the used score; nevertheless descriptions of commonly used scores can be found on the survey of Kitson et al. [2023, Sec. 4.1].

Our interest is in instances in which most local scores equal $-\infty$ and are not given as explicit input. Accordingly, for each node i we are given a collection of potential parent sets $\mathcal{C}_i$, the size of which can be substantially smaller than 2^{n-1}. The local scores $s_i(J)$ are only given for $J \in \mathcal{C}_i$. We let

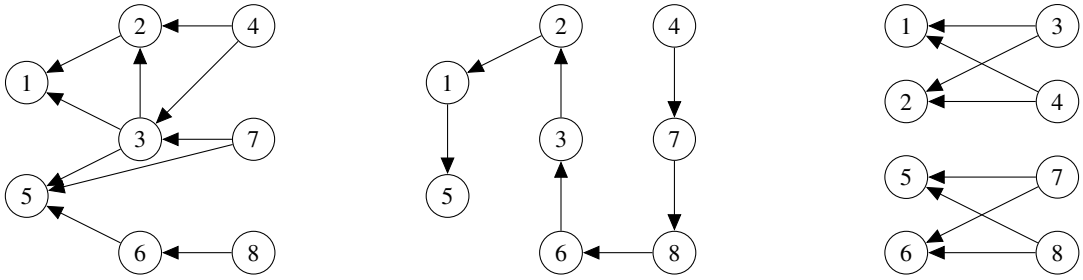
$$F := \sum_{i \in N} |\mathcal{C}_i|$$

denote the total size of the input.

In practice, potential parent sets are obtained using several ideas and combinations thereof. One is to include in $\mathcal{C}_i$ only sets that are contained in a relatively small set of candidate parents. Another idea is to only include sets whose cardinality does not exceed some given upper bound. A third technique is to exclude sets J for which there is a subset $J' \subseteq J$ with an equal or better local score, $s_i(J') \geq s_i(J)$; while this simple pruning rule may require computing the local scores for a large number of sets, more sophisticated analytic score bounds can also exclude sets without computing their scores [Correia et al., 2020]. Importantly, all these procedures result in collections $\mathcal{C}_i$ that are *closed under inclusion*, that, if $J \in \mathcal{C}_i$ and $J' \subseteq J$, then $J' \in \mathcal{C}_i$.

2.3 QUANTUM CIRCUITS AND QRAM

Quantum computation can be modeled by a quantum circuit that takes as input ℓ qubits representing the system's



(a) DAG A (b) Linear order L (c) Partial order P

Figure 1: Examples of a DAG, a linear order, and a partial order on the node set $\{1, 2, \ldots, 8\}$. We have $A \subseteq L$ and $P \subseteq L$; for example, $A_6 = \{8\}$, $L_6 = \{4, 7, 8\}$, and $P_6 = \{7, 8\}$. For the linear order, shown is its transitive reduction, i.e., only the edges necessary for determining the relation uniquely using transitivity. The partial order P is a member of the set of parallel bucket orders described in Section 4 (with $k = 4$).

initial state, then transforms the state by reversible quantum logic gates, until the final state of some $l \leq \ell$ qubits of interest is measured and an l-bit output is obtained. The additional power of quantum circuits in comparison to classical boolean circuits stems from the fact that ℓ qubits can represent a superposition (i.e., a linear combination) of all the 2^ℓ possible ℓ-bit vectors. The coefficients (i.e., amplitudes) encode a probability distribution over the possible vectors, a measurement returning the corresponding random variable. A quantum algorithm is thus a randomized algorithm. It has *bounded error* if, for all problem instances, the ouput is correct with probability at least $2/3$.

For classical algorithms, the boolean circuit model can yield pessimistic complexity bounds—for more practical settings, one assumes a random access memory (RAM).[1] Similarly, broader applicability of quantum computation is believed to require an equivalent *quantum RAM* (QRAM) [Giovannetti et al., 2008]. Importantly, QRAM enables invoking any time-T classical algorithm that uses RAM as a $O(T)$-time subroutine in a quantum algorithm.

2.4 QUANTUM SEARCH

Grover's algorithm [Grover, 1996], also known as quantum search, is a celebrated generic algorithm for finding a needle in a haystack. As described in the Introduction, it gives a quadratic speedup in relation to classical algorithms. We will make use of the following powerful extension:

Theorem 1 (Dürr and Høyer [1996], Ambainis et al. [2019])**.** *Suppose $f(x)$ is an integer computable for any given $x \in \{1, 2, \ldots, m\}$ by a bounded-error quantum algorithm in time T. Then there is a bounded-error quantum algorithm that computes $\max_{x=1}^{m} f(x)$ in time $O(T\sqrt{m}\log m)$.*

[1] An algorithm that runs in time T using a RAM can be simulated by a boolean circuit of size $T^2\mathrm{polylog}(T)$ [Cook and Reckhow, 1973, Pippenger and Fischer, 1979].

This result allows us to apply quantum search (i) in a maximization problem and (ii) recursively with only a negligible computational overhead.

3 FINDING A LINEAR ORDER

Various NP-hard graph problems can be viewed as finding an optimal node ordering. For our purposes it is convenient to consider the problem of computing

$$\max_{L} \sum_{i \in N} f(L_i, i)\,, \tag{1}$$

where the function f depends on the problem input and the maximization is over all linear orders L on N.

Ambainis et al. [2019] gave a quantum algorithm for any problem of that form:

Theorem 2 (Ambainis et al. [2019, Cor. 3.1])**.** *The problem (1) admits a bounded-error quantum algorithm that runs in time $O(1.817^n T)$, assuming f can be evaluated in time T.*

It is easy to see—and well known [Cooper and Herskovits, 1992, Eq. (9)]—that BNSL can be written in the above form by putting

$$f(L_i, i) := \max_{J \subseteq L_i : J \in \mathcal{C}_i} s_i(J)$$

Indeed, if A is an optimal DAG and L a topological ordering of its nodes, the score $s(A)$ is obtained as $\sum_{i \in N} f(L_i, i)$.

Since $f(L_i, i)$ can be computed in time $O(|\mathcal{C}_i|n)$ by a linear scan over the potential parent sets, we have got a quantum algorithm that solves BNSL in time $O(1.817^n F)$. To omit factors polynomial in n in the asymptotic bound, we used the fact that the constant base 1.817 of Theorem 2 was originally obtained by rounding up a strictly smaller constant.

But we can do better. We simply replace the classical linear scan by quantum search:

Theorem 3. *BNSL admits a bounded-error quantum algorithm that runs in time $O(1.817^n\sqrt{F})$.*

If F grows subexponentially in n, the bound can be simplified to $O(1.817^n)$. On the other hand, the base of the exponential exceeds the base 2 of the fastest classical algorithms already if $F = \Omega(1.212^n)$. In the next section, we give a different quantum algorithm that beats the known classical algorithms as long as $F = O(1.453^n)$.

4 COVERING BY PARTIAL ORDERS

Koivisto and Parviainen [2010] presented the following approach to a broad class of permutation problems, including ones of the form (1). Let $\mathcal{P}$ be a set of partial orders on N such that every linear order on N is an extension of at least one member in $\mathcal{P}$; we call $\mathcal{P}$ simply a *cover* on N. Now, for any function f of linear orders on L, we have

$$\max_L f(L) = \max_{P\in\mathcal{P}} \max_{L\supseteq P} f(L)$$

where the first maximization is over all linear orders on N. One example is when $\mathcal{P} = \{\emptyset\}$, rendering the outer maximization trivial. Another extreme case is when $\mathcal{P}$ consists of all linear orders on N, rendering the inner maximization trivial. In general, we have decomposed the original problem into $|\mathcal{P}|$ subproblems, each constrained by a different partial order.

In particular, we can write the BNSL problem as

$$\max_{P\in\mathcal{P}} g(P)\,,$$

with the subproblems

$$g(P) := \max_{L\supseteq P} \sum_{i\in N} \max_{J\subseteq L_i : J\in\mathcal{C}_i} s_i(J)\,. \tag{2}$$

Parviainen and Koivisto [2013] solved the subproblem by dynamic programming over the downsets of the partial order P. A *downset* is a subset $S \subseteq N$ that is closed under the relation, i.e., if $i \in S$ and $ji \in P$, then $j \in S$.

Proposition 4 (Parviainen and Koivisto [2013, Theorem 16])**.** *Suppose each $\mathcal{C}_i$ is closed under inclusion. Then the subproblem (2) admits an algorithm that runs in time $O(Dn^2 + Fn)$, where D is the number of downsets of P.*

We are now ready to apply quantum search over the cover $\mathcal{P}$. Combining Theorem 1 with the above result for the subproblem gives us a quantum algorithm for BNSL:

Proposition 5. *Let $\mathcal{P}$ be a cover on N, each $P \in \mathcal{P}$ having $O(D)$ downsets. Suppose each $\mathcal{C}_i$ is closed under inclusion and $\sum_i |\mathcal{C}_i| = O(D)$. Then BNSL admits a bounded-error quantum algorithm that runs in time $O(Dn^2|\mathcal{P}|^{1/2}\log|\mathcal{P}|)$.*

For simplicity, we here restricted the sizes of the sets $\mathcal{C}_i$ so that the running time for the subproblem is dominated by the number of downsets; this restriction will ease our further running time analysis, but is not crucial for the correctness of the algorithm.

Our goal is next to show that, with an appropriate choice of the cover $\mathcal{P}$, the running time is $O(c^n)$ for some constant c less than 2. Ignoring lower-order terms, our task is to minimize the product $D|\mathcal{P}|^{1/2}$.

Fortuitously, essentially the same task is already addressed by Koivisto and Parviainen [2010] in disguise: they aim at minimizing the space–time product, i.e., the product of the space complexity and the time complexity, which is given by $D^2|\mathcal{P}|$, again ignoring lower-order terms. (Both the space and the time requirement of classical dynamic programming over downsets scale roughly as D.)

They give the following construction of what they call *parallel bucket orders* (of length two). Suppose n is divisible by an even natural number k, which is a design parameter. Partition N arbitrarily into n/k sets $S_1, S_2, \ldots, S_{n/k}$ of size k. Let $\mathcal{P}$ consist of all partial orders on N of the form $R^1 \cup R^2 \cup \cdots \cup R^{n/k}$, where each R^t is a partial order on S_t obtained by splitting S_t into two subsets of size $k/2$ so that the elements in one set precede all other elements in the other set, i.e., $R^t = S' \times S''$ for some disjoint $S', S'' \subset S_t$ with $|S'| = |S''| = k/2$. See Fig. 1 for an illustration. Different values of k yield different space–time tradeoff. The product is minimized at $k = 26$, with the following numbers.

Proposition 6 (Koivisto and Parviainen [2010])**.** *Let N be an n-element set, with n divisible by 26. There is a cover $\mathcal{P}$ on N with $a^{n/26}$ members, each having $b^{n/26}$ downsets, where $a := \binom{26}{13}$ and $b := 2^{14} - 1$.*

The n-th root of $D|\mathcal{P}|^{1/2}$ is given by

$$a^{1/52} \cdot b^{1/26} < 1.3645 \cdot 1.4525 < 1.9820 =: c\,.$$

Since we round up the base c, the bound $O(c^n)$ suppresses any factor that grows subexponentially in n, including factors that arise when n is not divisible by 26 and the construction is modified accordingly (we omit details).

Theorem 7. *BNSL admits a bounded-error quantum algorithm that runs in time $O(1.982^n)$, provided that each $\mathcal{C}_i$ is closed under inclusion and $F = \sum_i |\mathcal{C}_i| = O(1.453^n)$.*

5 COMPUTATIONAL HARDNESS

In this section, we show that no classical algorithm can solve BNSL in time $O(c^n)$ with $c < 2$, assuming the following *strong exponential time hypothesis* (SETH) [Impagliazzo and Paturi, 2001, Impagliazzo et al., 2001].

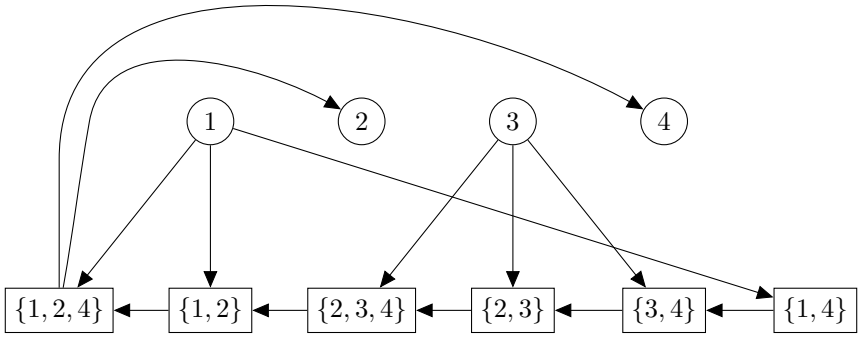


Figure 2: An optimal structure in a reduction from a 3-HITTING SET instance with a universe of 4 elements (circles) and a family of 6 sets (rectangles) to BNSL.

Hypothesis 8 (SETH)**.** *For any $\delta < 1$ there exists a number k such that the k-*CNF-SAT *problem over n variables cannot be solved in time $O\big(2^{\delta n}\big)$ by a classical algorithm.*

To connect the hardness of BNSL to SETH, we will construct a reduction from the k-HITTING SET problem: given a universe U of size n and a family $\mathcal{T}$ of subsets of U with at most k elements, is there a subset of U of size t that intersects all members of $\mathcal{T}$?

Theorem 9 (Cygan et al. [2016])**.** *If SETH holds, for any $\delta < 1$ there exists a number k such that the k-*HITTING SET *problem over a universe of size n cannot be solved in time $O\big(2^{\delta n}\big)$ by a classical algorithm.*

We state our result for BNSL in a form that replaces the parameter k above by the restriction that the input size is subexponential in the number of nodes. We leave it as an open problem to improve this to a polynomial bound.

Theorem 10. *If SETH holds, the* BNSL *problem over n variables and $2^{o(n)}$ potential parent sets cannot be solved in time $O\big(2^{\delta n}\big)$ for any $\delta < 1$ by a classical algorithm.*

Proof. Consider an instance $(U, \mathcal{T}, k, t)$ of the k-HITTING SET problem, where $U = \{u_1, u_2, \ldots, u_n\}$ is the universe and $\mathcal{T} = \{T_1, T_2, \ldots, T_m\}$ is a family of subsets of U of size at most k.

We first give a simpler reduction that results in a BNSL instance with $n + m$ nodes. Then, we continue by *sparsifying* the obtained instance by merging some of the nodes, rendering the number of nodes independent of m. Finally, we show that solving that instance in time $O\big(2^{\delta n}\big)$ for any $\delta < 1$ would break SETH.

We construct a BNSL instance where the nodes correspond to the n elements of the universe U and the m subsets in the given family $\mathcal{T}$; we denote these nodes with the same symbols u_i and T_j for notational convenience.

Define the following local scores (the rest being $-\infty$):

$$
\begin{aligned}
&s_{u_i}(\emptyset) = 0,\\
&s_{u_i}(\{T_1\}) = 1,\\
&s_{T_j}(\{u_i, T_{j+1}\}) = 0 \text{ if } u_i \in T_j \text{ and } j < m,\\
&s_{T_m}(\{u_i\}) = 0 \text{ if } u_i \in T_m.
\end{aligned}
$$

Suppose that H is a hitting set of $\mathcal{T}$. Then, the following parent set assignment is possible, that is, it yields a nonnegative score:

$$
\begin{aligned}
&A_{u_i} = \emptyset \text{ if } u_i \in H,\\
&A_{u_i} = \{T_1\} \text{ if } u_i \notin H,\\
&A_{T_j} = \{u_i, T_{j+1}\} \text{ for some } u_i \in H \text{ if } j < m,\\
&A_{T_m} = \{u_i\} \text{ for some } u_i \in H.
\end{aligned}
$$

Such a DAG attains a score $n - |H|$. An illustration is provided in Figure 2.

We claim that if H is a minimum-size hitting set for $\mathcal{T}$, then no DAG can exceed the score $n - |H|$. First, note that any DAG A with a nonnegative score corresponds to a hitting set

$$
H_A \coloneqq \{u_1, u_2, \ldots, u_n\} \cap \left(\bigcup_{j=1}^{m} A_{T_j} \right)
$$

for $\mathcal{T}$, since the parent set of each T_j must include a node u_i with $u_i \in T_j$ by the definition of the local scores. Further, the nodes $u_i \in H_A$ cannot have any parents, since otherwise this would violate acyclicity: the only potential non-empty parent set of u_i is $\{T_1\}$, but the DAG has to contain edges $T_1 \leftarrow T_2 \leftarrow \cdots \leftarrow T_j$ and $T_j \leftarrow u_i$ for some j. Finally, if $u_i \notin H_A$, then u_i has no children and can pick any of its potential parent sets. In particular, its local score is maximized by choosing $\{T_1\}$ with score 1. Thus, the BNSL instance admits a solution with score $n - t$ if and only there is a hitting set of size t for $\mathcal{T}$.

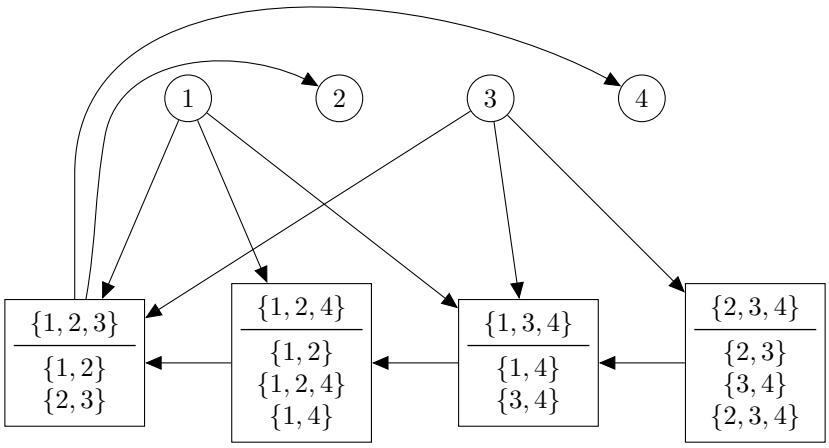


Figure 3: An optimal structure in a sparsified reduction from a 3-HITTING SET instance with a universe of 4 elements and a family of 6 sets to BNSL. Each set of the input is associated with at least one of the nodes, where the superset is written above the horizontal line and the associated subsets below it.

The constructed BNSL instance has $n + m$ nodes, which is too many to prove our theorem. We next sparsify the subset of m nodes that correspond to the members in $\mathcal{T}$.

We arbitrarily partition the universe U into $p \coloneqq \lceil n^{1/(k+1)} \rceil$ sets of (almost) equal size, $U_1, U_2, \ldots, U_p$, that is, their sizes $|U_i|$ differ by at most 1. For all $I \subseteq \{1, 2, \ldots, p\}$, let

$$U_I \coloneqq \bigcup_{i \in I} U_i \,.$$

Note that any $T \in \mathcal{T}$ is a subset of U_I for some I of size k.

Instead of introducing a node for each T_j in our BNSL instance, we introduce a node for each U_I with $|I| = k$. Label these sets arbitrarily by $T'_1, T'_2, \ldots, T'_{m'}$ with $m' = \binom{p}{k}$. For a subset P of U say that P *hits* T'_j if $P \subseteq T'_j$ and P intersects all $T \in \mathcal{T}$ with $T \subseteq T'_j$.

Define the following local scores (and potential parent sets):

$$\begin{aligned}
&s_{u_i}(\emptyset) = 0,\\
&s_{u_i}(\{T'_1\}) = 1,\\
&s_{T'_j}(P \cup \{T'_{j+1}\}) = 0 \text{ if } P \text{ hits } T'_j \text{ and } j < m',\\
&s_{T'_{m'}}(P) = 0 \text{ if } P \text{ hits } T'_{m'}.
\end{aligned}$$

In other words, the new local scores ensure that the parent set of T'_j hits all members of $\mathcal{T}$ that are its subsets.

As before, for any minimum-size hitting set H of $\mathcal{T}$, the maximum score $n - |H|$ is attained by the following parent set assignments:

$$\begin{aligned}
&A_{u_i} = \emptyset \text{ if } u_i \in H,\\
&A_{u_i} = \{T_1\} \text{ if } u_i \notin H,\\
&A_{T'_j} = (H \cap T'_j) \cup \{T'_{j+1}\} \text{ if } j < m',\\
&A_{T'_{m'}} = H \cap T'_{m'}.
\end{aligned}$$

This is illustrated in Figure 3.

The instance now contains $n + \binom{p}{k}$ nodes. Since $p = \lceil n^{1/(k+1)} \rceil$, we have $\binom{p}{k} \leq p^k = o(n)$.

To bound the number of potential parent sets, observe that each node T'_j has at most $2^{|T'_j|}$ potential parent sets. Since T'_j is a union of k parts U_i, each part of size at most $\lceil n/p \rceil \leq \lceil n^{k/(k+1)} \rceil = o(n)$, the number of potential parent sets of T'_j is at most $2^{k\lceil n/p \rceil} = 2^{o(n)}$. The total number of potential parent sets is thus bounded by

$$2n + \binom{p}{k} 2^{o(n)} = 2^{o(n)} \,.$$

In summary, constructing the instance takes subexponential time, there are subexponentially many potential parent sets, and the number of nodes is asymptotically equivalent to n.

Assume now that SETH holds but any instance of BNSL with n' nodes could be solved in time $O(2^{\delta' n'})$ for some $\delta' < 1$. Put $\delta \coloneqq (\delta' + 1)/2 < 1$. By Theorem 9, there exists a k such that k-HITTING SET with a universe of size n cannot be solved in time $O\big(2^{\delta n}\big)$. However, we showed that any such instance can be reduced to an instance of BNSL on $n' = n + o(n)$ variables in time $2^{o(n)}$. By our assumption, we can solve it in time $O(2^{\delta' \cdot (n + o(n))}) = O(2^{o(n)} 2^{\delta' n}) = O(2^{\delta n})$, which is a contradiction. □

6 CONCLUDING REMARKS

We have shown quantum speedups for the problem of Bayesian network structure learning. Our two algorithms

are built on rather sophisticated previous results: a quantum algorithm for a related problem [Ambainis et al., 2019] and a classical algorithm for the same problem [Parviainen and Koivisto, 2013]. On the other hand, the ways we employed these previous results are technically relatively simple. We also proved that similar speedups presumably are not possible for classical algorithms, suggesting that the BNSL problem admits a "quantum advantage."[2]

An obvious question for further research is whether there are significantly faster quantum algorithms, e.g., ones with running time close to $O(2^{n/2})$ or others that yield a quantum speedup even when we do not bound the number of potential parent sets. Achieving the former target would most likely imply a new algorithm for the travelling salesman problem that beats current time bound of $O(1.728^n)$ [Ambainis et al., 2019]. The latter question assumes that the local scores are given implicitly, which is not an obstacle per se, as the local score of a given node and parent set can be computed efficiently from data for commonly used scoring functions.

Our algorithms may not have practical value in the near future. The speedup factor $(2/1.817)^n$ of our first algorithm achieves 10 at $n \approx 24$ and 100 at $n \approx 48$. However, the hidden subexponential factors are likely to favor the classical algorithms in practice even if the number of potential parent sets F is small, say, cubic in n. Perhaps most importantly, our algorithms rely on QRAM, of which size is exponential in n. While different QRAM architectures have been proposed [Giovannetti et al., 2008, Park et al., 2019], there is no physical realization of the ideas yet. Currently we do not know whether the role of QRAM is critical for achieving any polynomial quantum speedup.

Regarding lower bounds for classical algorithms, there are several directions for future research. Theorem 10 does not rule out a faster algorithm when the maximum size of any potential parent set is bounded by a constant; the problem is solvable in polynomial time when the maximum size is one [Chu and Liu, 1965, Edmonds, 1967], but for larger upper bounds we only know that the problem is NP-hard [Chickering, 1995]. One could also attempt to prove conditional lower bounds under some other established hypothesis not known to be implied by SETH such as the set cover conjecture [Cygan et al., 2015, p. 507].

Acknowledgements

Research partially supported by the Research Council of Finland, grants 316771 and 351156.

[2]We use scare quotes because some researchers and practitioners reserve the term advantage for speedups that are exponential or experimentally demonstrated.

References

Andris Ambainis, Kaspars Balodis, Janis Iraids, Martins Kokainis, Krisjanis Prusis, and Jevgenijs Vihrovs. Quantum speedups for exponential-time dynamic programming algorithms. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019*, pages 1783–1793. SIAM, 2019.

Mark Bartlett and James Cussens. Integer linear programming for the Bayesian network structure learning problem. *Artif. Intell.*, 244:258–271, 2017.

Richard Bellman. Dynamic programming treatment of the travelling salesman problem. *J. ACM*, 9(1):61–63, 1962.

Hans L. Bodlaender, Fedor V. Fomin, Arie M. C. A. Koster, Dieter Kratsch, and Dimitrios M. Thilikos. A note on exact algorithms for vertex ordering problems on graphs. *Theory Comput. Syst.*, 50(3):420–432, 2012.

David M. Chickering. Learning Bayesian networks is NP-complete. In *Learning from Data: Fifth International Workshop on Artificial Intelligence and Statistics, AISTATS 1995*, pages 121–130. Springer, 1995.

Yeong-Jin Chu and Tseng-Hong Liu. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14:1396–1400, 1965.

Stephen A. Cook and Robert A. Reckhow. Time bounded random access machines. *J. Comput. Syst. Sci.*, 7(4): 354–375, 1973.

Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Mach. Learn.*, 9:309–347, 1992.

Alvaro Henrique Chaim Correia, James Cussens, and Cassio P. de Campos. On pruning for score-based Bayesian network structure learning. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020*, volume 108 of *Proceedings of Machine Learning Research*, pages 2709–2718. PMLR, 2020.

Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

Marek Cygan, Holger Dell, Daniel Lokshtanov, Dániel Marx, Jesper Nederlof, Yoshio Okamoto, Ramamohan Paturi, Saket Saurabh, and Magnus Wahlström. On problems as hard as CNF-SAT. *ACM Trans. Algorithms*, 12 (3), 2016.

Evgeny Dantsin, Vladik Kreinovich, and Alexander Wolpert. On quantum versions of record-breaking algorithms for SAT. *SIGACT News*, 36(4):103–108, 2005.

Vedran Dunjko, Jacob M. Taylor, and Hans J. Briegel. Quantum-enhanced machine learning. *Phys. Rev. Lett.*, 117:130501, 2016.

Christoph Dürr and Peter Høyer. A quantum algorithm for finding the minimum. *CoRR*, quant-ph/9607014, 1996.

Jack Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71(4):233–240, 1967.

Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Quantum random access memory. *Phys. Rev. Lett.*, 100: 160501, 2008.

Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, STOC 1996*, pages 212–219. ACM, 1996.

Niels Grüttemeier and Christian Komusiewicz. Learning Bayesian networks under sparsity constraints: A parameterized complexity analysis. *J. Artif. Intell. Res.*, 74: 1225–1267, 2022.

Michael Held and Richard M. Karp. A dynamic programming approach to sequencing problems. In *Proceedings of the 16th ACM national meeting, ACM 1961*, page 71. ACM, 1961.

Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

Ashish Kapoor, Nathan Wiebe, and Krysta M. Svore. Quantum perceptron models. In *Advances in Neural Information Processing Systems 29, NeurIPS 2016*, pages 3999–4007, 2016.

Neville Kenneth Kitson, Anthony C. Constantinou, Zhigao Guo, Yang Liu, and Kiattikun Chobtham. A survey of Bayesian network structure learning. *Artif. Intell. Rev.*, 56(8):8721–8814, 2023.

Mikko Koivisto and Pekka Parviainen. A space-time tradeoff for permutation problems. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010*, pages 484–492. SIAM, 2010.

Mikko Koivisto and Antti Röyskö. Fast multi-subset transform and weighted sums over acyclic digraphs. In *Proceedings of the 17th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2020*, volume 162 of *LIPIcs*, pages 29:1–29:12. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models - Principles and Techniques*. MIT Press, 2009.

Sascha Ott, Seiya Imoto, and Satoru Miyano. Finding optimal models for small gene networks. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 557–567, 2004.

Bryan O'Gorman, Ryan Babbush, Alejandro Perdomo-Ortiz, Alán Aspuru-Guzik, and Vadim Smelyanskiy. Bayesian network structure learning using quantum annealing. *Eur. Phys. J. Spec. Top.*, 224:163–188, 2015.

Daniel K. Park, Francesco Petruccione, and June-Koo Kevin Rhee. Circuit-based quantum random access memory for classical data. *Scientific Reports*, 9(1), 2019.

Pekka Parviainen and Mikko Koivisto. Finding optimal Bayesian networks using precedence constraints. *J. Mach. Learn. Res.*, 14(1):1387–1415, 2013.

Nicholas Pippenger and Michael J. Fischer. Relations among complexity measures. *J. ACM*, 26(2):361–381, 1979.

Mathieu Roget, Giuseppe Di Molfetta, and Hachem Kadri. Quantum perceptron revisited: Computational-statistical tradeoffs. In *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence, UAI 2022*, volume 180 of *Proceedings of Machine Learning Research*, pages 1697–1706. PMLR, 2022.

Tomi Silander and Petri Myllymäki. A simple approach for finding the globally optimal Bayesian network structure. In *Proceedings of the 22nd Conference in Uncertainty in Artificial Intelligence, UAI 2006*. AUAI Press, 2006.

Ajit Singh and Andrew Moore. Finding optimal Bayesian networks by dynamic programming. Technical report, Carnegie Mellon University, School of Computer Science, 2005.

Vicente P. Soloviev, Concha Bielza, and Pedro Larrañaga. Quantum approximate optimization algorithm for Bayesian network structure learning. *Quantum Inf. Process.*, 22(1):19, 2023.

Jin Tian and Ru He. Computing posterior probabilities of structural features in Bayesian networks. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2009*, pages 538–547. AUAI Press, 2009.

Changhe Yuan and Brandon M. Malone. Learning optimal Bayesian networks: A shortest path perspective. *J. Artif. Intell. Res.*, 48:23–65, 2013.