



# On the Representation of Boolean and Real Functions as Hamiltonians for Quantum Computing

STUART HADFIELD, Quantum Artificial Intelligence Laboratory at NASA Ames Research Center, USRA Research Institute for Advanced Computer Science, and Department of Computer Science, Columbia University

Mapping functions on bits to Hamiltonians acting on qubits has many applications in quantum computing. In particular, Hamiltonians representing Boolean functions are required for applications of quantum annealing or the quantum approximate optimization algorithm to combinatorial optimization problems. We show how such functions are naturally represented by Hamiltonians given as sums of Pauli Z operators (Ising spin operators) with the terms of the sum corresponding to the function's Fourier expansion. For many classes of Boolean functions which are given by a compact description, such as a Boolean formula in conjunctive normal form that gives an instance of the satisfiability problem, it is #P-hard to compute its Hamiltonian representation, i.e., as hard as computing its number of satisfying assignments. On the other hand, no such difficulty exists generally for constructing Hamiltonians representing a real function such as a sum of local Boolean clauses each acting on a fixed number of bits as is common in constraint satisfaction problems. We show composition rules for explicitly constructing Hamiltonians representing a wide variety of Boolean and real functions by combining Hamiltonians representing simpler clauses as building blocks, which are particularly suitable for direct implementation as classical software. We further apply our results to the construction of controlled-unitary operators, and to the special case of operators that compute function values in an ancilla qubit register. Finally, we outline several additional applications and extensions of our results to quantum algorithms for optimization. A goal of this work is to provide a *design toolkit for quantum optimization* which may be utilized by experts and practitioners alike in the construction and analysis of new quantum algorithms, and at the same time to provide a unified framework for the various constructions appearing in the literature.

CCS Concepts: • **Theory of computation** → *Quantum computation theory; Design and analysis of algorithms;*  
• **Mathematics of computing** → *Discrete mathematics; Combinatorial optimization;*

We thank Al Aho for support and guidance, and the members of the Quantum AI Lab for providing helpful comments and suggestions. This work was initiated thanks to the support of the USRA Feynman Quantum Computing Academy summer internship program and NASA Ames Research Center, and further developed while at Columbia University. S.H. was additionally supported by NASA Academic Mission Services, Contract No. NNA16BD14C. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purpose not withstanding any copyright annotation thereon.

Author's address: S. Hadfield, Quantum Artificial Intelligence Laboratory at NASA Ames Research Center, Moffett Field, California, 94035, USRA Research Institute for Advanced Computer Science, Mountain View, California, 94043, Department of Computer Science, Columbia University, New York, New York, 10027; emails: [stuart.hadfield@nasa.gov](mailto:stuart.hadfield@nasa.gov), [shadfield@usra.edu](mailto:shadfield@usra.edu).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).

© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.

2643-6817/2021/12-ART18 \$15.00

<https://doi.org/10.1145/3478519>

Additional Key Words and Phrases: Quantum optimization, Boolean and pseudo-Boolean functions, diagonal Hamiltonians, Fourier analysis on the Boolean cube

#### ACM Reference format:

Stuart Hadfield. 2021. On the Representation of Boolean and Real Functions as Hamiltonians for Quantum Computing. *ACM Trans. Quantum Comput.* 2, 4, Article 18 (December 2021), 21 pages.

<https://doi.org/10.1145/3478519>

## 1 INTRODUCTION

A basic requirement of many quantum algorithms is the ability to translate between mathematical functions acting on a domain, typically strings of bits, and quantum mechanical Hamiltonian operators acting on qubits. In particular, mapping Boolean or real functions to Hamiltonians has important applications in quantum algorithms and heuristics for solving decision or optimization problems such as quantum annealing and adiabatic quantum optimization [22, 34, 42], or the quantum approximate optimization algorithm and **quantum alternating operator ansatz (QAOA)** [21, 28, 33], or the related variational quantum eigensolver [50]. Explicit Hamiltonian constructions for the application of these algorithms to a variety of prototypical problems can be found in [28, 40], though prior work has mostly focused on reductions to quadratic Hamiltonians at the expense of additional qubits, for example, in the penalty term approach of quantum annealing; we explain how direct (not necessarily quadratic) mappings are desirable for quantum gate model algorithms. Such quantum algorithms are promising, in particular, as possible paths towards performing useful computation on near-term quantum computing devices. Indeed, decision and optimization problems are ubiquitous across science and engineering, yet often appear to be computationally difficult. Despite years of investigation, efficient algorithms often remain elusive [3, 4]. Hence, the potential for new approaches to tackling these problems on quantum computers is an exciting development.

Nevertheless, the conceptual barrier to entry to studying these quantum algorithms and providing new insights remains high, especially for practitioners in the domain where a given problem arises, who may not be familiar with quantum computing beyond the basics. It is thus important to develop tools and methodologies, which are accessible to scientists and researchers from different domains, and are as independent of knowing the low-level details of quantum computing as possible, towards enabling easier cross-fertilization of different ideas and techniques. At the same time, it is useful to provide a rigorous general foundation for existing constructions and tools found in the literature, often in a specific context. Thus, a motivating goal of this work is to provide a *design toolkit* of basic results and methodologies which can be used by experts or laymen alike to design, implement, and analyze quantum algorithms. To this end, our results allow for straightforward implementation as computer programs such that Hamiltonian and quantum circuit mappings for many classes of problems may be automatically generated.

In this article, we show a general theory of mappings of Boolean and real functions to diagonal Hamiltonians acting on qubits, and give simple logical rules for the explicit construction of these Hamiltonians which include many common classes of functions such as Boolean formulas and circuits. We also address the question of when such Hamiltonians may or may not be constructed efficiently. We show how our results may be applied to the construction of unitary operators controlled by Boolean predicates, which are used, for example, in several of the QAOA mixing operator constructions of [28]. Our results are general and give a methodical approach to derive many of the mappings in the literature such as those of [28, 40]. We emphasize that our results

have applications to quantum algorithms beyond quantum annealing or QAOA, and we discuss a number of examples.

We elaborate on our results, which we summarize in the next section. Consider a function  $f$  acting on  $n$  bits. We say a Hamiltonian  $H_f$  *represents*  $f$  if it satisfies

$$H_f|x\rangle = f(x)|x\rangle, \quad (1)$$

for each input string  $x \in \{0, 1\}^n$  with corresponding (i.e., encoded as) computational basis state  $|x\rangle$ . (We will typically assume  $f$  is decidable and moreover can be efficiently evaluated classically; see, for example, [45] for a discussion of Hamiltonian families that encode the halting problem.) We show how arbitrary  $n$ -bit Boolean or real functions are naturally represented as diagonal Hamiltonians given by weighted sums of Pauli  $Z$  operators, with terms corresponding to the function's Fourier expansion, as summarized in Theorem 1 below. Such Hamiltonians generalize the Ising model of interacting spin-1/2 particles well-known in physics. Our results rely on the Fourier analysis of Boolean functions, which has a long history of important applications in computer science [8, 27, 30, 35, 39, 46] and in particular quantum computing [1, 7, 12, 43]. We use our results to derive explicit Hamiltonian representations of a number of basic Boolean predicates shown in Table 1 below. Combining the rules of classical propositional logic with the properties of the Fourier expansion leads to composition rules for constructing Hamiltonians representing conjunctions, disjunctions, exclusive or, and other functions of simpler clauses by combining their Hamiltonian representations in particular ways; see Theorem 2 below. Furthermore, these mappings directly extend to constructing Hamiltonians representing weighted sums of clauses, which are a primary class of Hamiltonians considered in quantum annealing and QAOA, for example, for constraint satisfaction problems.

We also consider the computational complexity of such Hamiltonian constructions, which naturally depends on how the function  $f$  is provided as input. Many combinatorial properties of a given function can be “read off” from its Fourier coefficients [48]. This presents an obstruction to computing the Hamiltonian representation for general Boolean functions; we show that computing the identity component of  $H_f = \widehat{f}(\emptyset)I + \dots$ , which is given by the first Fourier coefficient  $\widehat{f}(\emptyset)$  of  $f$ , is as hard as counting the number of inputs such that  $f = 1$ , which in general is computationally intractable. For example, if  $f$  is a Boolean formula on  $n$  variables given in conjunctive normal formula form with a poly( $n$ ) size description, i.e., an instance of the **satisfiability** problem (**SAT**), then this task is #P-hard and hence it is not believed possible to efficiently compute  $\widehat{f}(\emptyset)$  in general; if such a classical polynomial-time algorithm existed then we would have P=NP. Hence, we cannot efficiently construct explicit Hamiltonian representations of many  $n$ -bit Boolean functions, even when such a function may be efficiently evaluated. We apply our results to show a similar result holds for computing the Pauli operator expansion of quantum circuits that compute  $f$  in a register.

Nevertheless, there is no such difficulty for *local* Boolean functions  $f_j$  where each  $f_j$  acts on a constant number of bits. This allows us to efficiently construct Hamiltonians representing *pseudo-Boolean* functions of the form  $f(x) = \sum_{j=1}^m w_j f_j(x)$ , with  $w_j \in \mathbb{R}$  and  $m = \text{poly}(n)$ . Such real functions (and their corresponding Hamiltonians) may be constructed to directly encode optimization problems of interest, or such that their minimum value arguments (ground state eigenvectors) encodes the solution to a corresponding decision problem; similar to, for example, how solving the MAX-SAT problem (finding the maximum possible number of satisfied clauses) also solves SAT. For a pseudo-Boolean function, its Fourier coefficients do not allow its extremal values to be “read off” in the same way and so its Hamiltonian representation can often be computed efficiently. Indeed, this is a common approach to encoding decision problems such as SAT into the framework

Table 1. Hamiltonians Representing Basic Boolean Clauses

$f(x)$	$H_f$	$f(x)$	$H_f$
$x$	$\frac{1}{2}I - \frac{1}{2}Z$	$\bar{x}$	$\frac{1}{2}I + \frac{1}{2}Z$
$x_1 \oplus x_2$	$\frac{1}{2}I - \frac{1}{2}Z_1Z_2$	$\bigoplus_{j=1}^k x_j$	$\frac{1}{2}I - \frac{1}{2}Z_1Z_2 \dots Z_k$
$x_1 \wedge x_2$	$\frac{1}{4}I - \frac{1}{4}(Z_1 + Z_2 - Z_1Z_2)$	$\bigwedge_{j=1}^k x_j$	$\frac{1}{2^k} \prod_j (I - Z_j)$
$x_1 \vee x_2$	$\frac{3}{4}I - \frac{1}{4}(Z_1 + Z_2 + Z_1Z_2)$	$\bigvee_{j=1}^k x_j$	$I - \frac{1}{2^k} \prod_j (I + Z_j)$
$\overline{x_1 x_2}$	$\frac{3}{4}I + \frac{1}{4}(Z_1 + Z_2 - Z_1Z_2)$	$x_1 \Rightarrow x_2$	$\frac{3}{4}I + \frac{1}{4}(Z_1 - Z_2 + Z_1Z_2)$

of quantum annealing [40]. Our results likewise apply to designing penalty term approaches for problems or encodings with hard feasibility constraints, which we discuss in Section 3.1.

### 1.1 Main Results: A Toolkit for Quantum Optimization Algorithms

We summarize our main results. The details of representing Boolean and real (pseudo-Boolean) functions as Hamiltonians are addressed in Section 2 where we first briefly review and then apply Fourier analysis on the Boolean cube. In Section 3, we outline the application of our results to constructing diagonal operators for optimization applications, including ground state logic approaches and the implementation of diagonal phase unitaries for QAOA. We extend our results to (non-diagonal) controlled unitaries in Section 4, including the setting of operators computing Boolean functions in ancilla registers, which is complementary to our main Hamiltonian approach. Several illustrative examples and remarks are provided throughout. We conclude with a discussion of our results and future research questions in Section 5.

**1.1.1 Boolean Functions.** Boolean-valued functions on the Boolean cube  $\{0, 1\}^n$  are uniquely represented as diagonal Hamiltonians as in (1) by a real multi-linear polynomial of Pauli  $Z$  operators, with terms corresponding to the function's Fourier expansion.

**THEOREM 1.** *For a Boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ , the unique Hamiltonian on  $n$  qubits satisfying  $H_f|x\rangle = f(x)|x\rangle$  for each computational basis state  $|x\rangle$  is*

$$H_f = \sum_{S \subseteq [n]} \widehat{f}(S) \prod_{j \in S} Z_j = \widehat{f}(\emptyset)I + \sum_{j=1}^n \widehat{f}(\{j\})Z_j + \sum_{j < k} \widehat{f}(\{j, k\})Z_jZ_k + \dots, \quad (2)$$

where the Fourier coefficients

$$\widehat{f}(S) = \frac{1}{2^n} \sum_{x \in \{0, 1\}^n} f(x)(-1)^{S \cdot x} = \frac{1}{2^n} \text{tr} \left( H_f \prod_{j \in S} Z_j \right), \quad (3)$$

satisfy  $\widehat{f}(\emptyset) \in [0, 1]$ ,  $\widehat{f}(S) \in [-\frac{1}{2}, \frac{1}{2}]$  for  $S \neq \emptyset$ ,

$$\sum_{S \subseteq [n]} \widehat{f}(S) = f(0^n), \quad (4)$$

where  $0^n$  denotes the input bit string of all 0s, and

$$\sum_{S \subseteq [n]} \widehat{f}(S)^2 = \frac{1}{2^n} \sum_{x \in \{0, 1\}^n} f(x) = \widehat{f}(\emptyset). \quad (5)$$

The proof of the theorem is shown in Section 2.1. Here we have used the notation  $S \cdot x := \sum_{j \in S} x_j$ ,  $[n] := \{1, 2, \dots, n\}$ , and  $Z_j := I \otimes \dots \otimes I \otimes Z \otimes I \dots \otimes I$  to denote the Pauli  $Z$  operator applied to the  $j$ th qubit. We emphasize that the theorem also applies to functions that depend only on a subset of

Table 2. Example: Hamiltonians Representing Boolean Clauses on three Variables

$f(x)$	$H_f$
$MAJ(x_1, x_2, x_3)$	$\frac{1}{2}I - \frac{1}{4}(Z_1 + Z_2 + Z_3 - Z_1Z_2Z_3)$
$NAE(x_1, x_2, x_3)$	$\frac{3}{4}I - \frac{1}{4}(Z_1Z_2 + Z_1Z_3 + Z_2Z_3)$
$MOD_3(x_1, x_2, x_3)$	$\frac{1}{4}I + \frac{1}{4}(Z_1Z_2 + Z_2Z_3 + Z_1Z_3)$
$1in3(x_1, x_2, x_3)$	$\frac{1}{8}(3I + Z_1 + Z_2 + Z_3 - Z_1Z_2 - Z_2Z_3 - Z_1Z_3 - 3Z_1Z_2Z_3)$

$k < n$  bits, where  $k$  may be independent of  $n$ . For example, the function  $x_1 \oplus x_2 \oplus x_3$  is easily seen to map to the Hamiltonian  $\frac{1}{2}I - \frac{1}{2}Z_1Z_2Z_3$ . We note that expansions similar to (2) have been considered in the context of implementing diagonal unitaries [51, 56] and analyzing their quantum gate complexity [2], whereas Theorem 1 and our results to follow are more generally applicable. The generalization of Fourier analysis to functions over complex numbers (i.e., quantum circuit amplitudes) has been employed to analyze the complexity of sampling from quantum circuit models such as IQP circuits [12, 15]. We emphasize that in gate model applications, higher order Pauli  $Z$  interactions may be implemented efficiently; this is in contrast to quantum annealing applications where interactions must typically be reduced to low-order (quadratic) ones using ancillary qubits to accommodate physical implementation [40]. See Section 3.3 for additional discussion.

Thus, from (5) we see that computing the Hamiltonian representation (2) of a Boolean function is equivalent to computing its Fourier expansion, and is at least as computationally difficult as computing  $\hat{f}(\emptyset)$ .

**COROLLARY 1.** *Computing the identity coefficient  $\hat{f}(\emptyset)$  of the Hamiltonian  $H_f$  representing a Boolean SAT formula  $f$  (given in conjunctive normal form) is #P-hard. Deciding if  $\hat{f}(\emptyset) = 0$  is equivalent to deciding if  $f$  is unsatisfiable, in which case  $H_f$  reduces to the 0 matrix.*

Hence, given a Boolean function  $f$  such that counting its number of satisfying inputs is #P-hard, computing the identity coefficient  $\hat{f}(\emptyset)$  of its Hamiltonian representation will be #P-hard also. Such hard counting problems include not only functions corresponding to NP-hard decision problems such as SAT, but also certain functions corresponding to decision problems decidable in polynomial time, such as counting the number of perfect matchings in a bipartite graph; see, e.g., [3]. We emphasize that even if we can compute the value of each Fourier coefficient, a Hamiltonian  $H_f$  representing a general Boolean or real function on  $n$  bits may require a number of Pauli  $Z$  terms that is exponentially large with respect to  $n$ ; such an example is the logical AND of  $n$  variables (see Table 1).

On the other hand, when a Boolean clause  $f$  acts only a number of bits  $k < n$  that is constant or logarithmically scaling we may always efficiently construct its Hamiltonian representation as the number of non-zero terms in the sum (2) (the size of  $H_f$ ) is in this case at most  $2^k$ . The degree of  $H_f$ ,  $\deg(H_f) = \deg(f) = d$ , is the maximum locality (number of qubits acted on) of any such term. Bounded-degree Hamiltonians  $H_f = \sum_{S \subseteq [k], |S| \leq d} \hat{f}(S) \prod_{j \in S} Z_j$ ,  $k \leq n$ , are similarly always efficiently representable<sup>1</sup> as we show  $\text{size}(H_f) \leq (e/d)^{d-1}k^d + 1$  which is always polynomial in  $n$  if  $d = O(1)$ . For example, bounded-locality constraint satisfaction problems such as MAX-CUT or MAX- $\ell$ -SAT (up to  $\ell = O(\log n)$ ) are described by sums of local clauses and hence are always

<sup>1</sup>We say a family of functions  $\{f_{n,j}\}$  on  $n$  bits is *efficiently representable* as the Hamiltonians  $\{H_{f_{n,j}}\}$  if  $\text{size}(H_{f_{n,j}})$  grows polynomially with  $n$ .

efficiently representable as Hamiltonians. We summarize mappings of some important basic clauses in Table 1 above.

**1.1.2 Constructing Hamiltonians using Propositional Logic.** We show formal rules for combining Hamiltonians representing different Boolean functions to obtain Hamiltonians representing more complicated logical expressions. In particular, we consider the logical negation ( $\neg$ ), conjunction ( $\wedge$ ), disjunction ( $\vee$ ), exclusive or ( $\oplus$ ), and implication ( $\Rightarrow$ ) operations, and addition or multiplication by a real number when Boolean functions are embedded as a subset of real-valued functions.

**THEOREM 2 (COMPOSITION RULES).** *Let  $f, g$  be Boolean functions represented by Hamiltonians  $H_f, H_g$ . Then the Hamiltonians representing basic operations on  $f$  and  $g$  are given by*

$$\begin{aligned} - H_{\neg f} &= H_{\bar{f}} = I - H_f & - H_{f \Rightarrow g} &= I - H_f + H_f H_g \\ - H_{f \wedge g} &= H_{fg} = H_f H_g & - H_{f \vee g} &= H_f + H_g - H_f H_g \\ - H_{f \oplus g} &= H_f + H_g - 2H_f H_g & - H_{af+bg} &= aH_f + bH_g \quad a, b \in \mathbb{R}. \end{aligned}$$

The proof of the theorem is given in Section 2.2. Hamiltonians for a wide variety of functions can be easily constructed using the composition rules and results for basic clauses as in Table 1, in particular Boolean formulas and circuits.

**Remark 1.** Theorems 1, 2, and 3 below facilitate straightforward software implementation for generating diagonal Hamiltonians, for example, automating QAOA mappings for constraint satisfaction problems with increasingly general types of constraints. Examples of three variable clauses are generated in Table 2. Such Hamiltonians may be useful *intermediate representations* in applications.

**1.1.3 Pseudo-Boolean Functions.** Consider a real function  $f$  on  $n$  bits given as a weighted sum of Boolean functions  $f_j$ ,

$$f(x) = \sum_{j=1}^m w_j f_j(x) \quad w_j \in \mathbb{R},$$

where each  $f_j$  acts on a subset of the  $n$  bits, and in the applications we consider often  $m = \text{poly}(n)$ . Objective functions for constraint satisfaction problems, considered, for example, in QAOA, are often expressed in this form, with each  $f_j$  given by a Boolean clause. A different example is the penalty term approach of quantum annealing, where the objective function is augmented with a number of high-weight penalty terms which perform (typically, local) checks that a state is valid; see Section 3.1 for a discussion. For such *pseudo-Boolean functions* we have the following result generalizing Thm. 1, which is shown in Section 2.3. Pseudo-Boolean optimization is a rich topic and we refer the reader to [13] for an overview.

**THEOREM 3.** *For an  $n$ -bit real function  $f : \{0, 1\}^n \rightarrow \mathbb{R}$  the unique Hamiltonian on  $n$  qubits satisfying  $H_f|x\rangle = f(x)|x\rangle$  is*

$$H_f = \sum_{S \subseteq [n]} \widehat{f}(S) \prod_{j \in S} Z_j, \quad (6)$$

with coefficients  $\widehat{f}(S) = \frac{1}{2^n} \sum_{x \in \{0, 1\}^n} f(x) (-1)^{S \cdot x} = \frac{1}{2^n} \text{tr}(H_f \prod_{j \in S} Z_j) \in \mathbb{R}$ .

*In particular, for a pseudo-Boolean function  $f(x) = \sum_{j=1}^m w_j f_j(x)$ ,  $w_j \in \mathbb{R}$ , where the  $f_j$  are Boolean functions corresponding to Hamiltonians  $H_{f_j}$  as in (2), we have*

$$H_f = \sum_{j=1}^m w_j H_{f_j} \quad \text{and} \quad \widehat{f}(S) = \sum_j \widehat{f_j}(S) \in \mathbb{R}, \quad (7)$$

with  $d := \deg(H_f) \leq \max_j \deg(f_j)$  and  $\text{size}(H_f) \leq \min\{\sum_j \text{size}(H_{f_j}), (e/d)^{d-1} n^d + 1\}$ .



Our results also yield direct cost estimates for quantum simulation of diagonal Hamiltonians, which we elaborate on in Section 3.3, though we emphasize that simulation is not the motivating application our results; nevertheless, it is an important piece of our design toolkit. Indeed, such operators occur, for example, in QAOA or Grover's algorithm. As the terms in (6) mutually commute, we can *simulate* such an a Hamiltonian, i.e., implement the operator  $U = \exp(-iH_f t)$  for some fixed  $t \in \mathbb{R}$ , using  $O(\deg(H_f) \cdot \text{size}(H_f))$  many basic quantum gates. Indeed, similar constructions for implementing diagonal unitaries have been previously proposed [51, 56], though more efficient circuits often result by utilizing ancilla qubits such as the operators we consider in Proposition 2 below [6, 17].

The remaining two items of the section demonstrate how our results may be applied to the construction of more general (non-diagonal) Hamiltonians and unitary operators.

**1.1.4 Controlled Hamiltonians and Unitaries.** In many applications we require controlled Hamiltonian simulations. Such operators are closely related to *block encodings* common in quantum computing [16]. Consider two quantum registers of  $k + n$  qubits. Given a Boolean function  $f(y)$  acting on  $k$  bits and a unitary operator  $U$  acting on  $n$  qubits, we define the  $(k + n)$ -qubit  $f$ -controlled unitary operator  $\Lambda_f(U)$  by its action on computational basis states

$$\Lambda_f(U)|y\rangle|x\rangle = \begin{cases} |y\rangle|x\rangle & f(y) = 0 \\ |y\rangle U|x\rangle & f(y) = 1. \end{cases}$$

Equivalently, as  $H_f + H_{\bar{f}} = I$  we have the useful decomposition  $\Lambda_f(U) = H_f \otimes U + H_{\bar{f}} \otimes I$ .

If  $U$  is self-adjoint, then  $\Lambda_f(U)$  is also a Hamiltonian. When  $U$  is given as a time evolution under a Hamiltonian  $H$  for a time  $t$ , we have the following.

**PROPOSITION 1.** *Let  $f$  be a Boolean function represented by a  $k$ -qubit Hamiltonian  $H_f$ , and let  $H$  be an arbitrary Hamiltonian acting on  $n$  disjoint qubits. Then the  $(k + n)$ -qubit Hamiltonian*

$$\tilde{H}_f = H_f \otimes H, \tag{8}$$

*corresponds to  $f$ -controlled evolution under  $H$ , i.e., for  $t \in \mathbb{R}$  satisfies*

$$e^{-i\tilde{H}_f t} = \Lambda_f(e^{-iHt}). \tag{9}$$

The proof follows from from exponentiating (8) directly, see Section 4.1. We emphasize the Proposition may be applied to generic control functions beyond the AND functions commonly considered in the literature (e.g., multi-controlled Toffoli gates).

**Remark 2 (Advanced Mixing Operators for QAOA).** Proposition 1 may be applied together with the Theorems above to design controlled mixing operators for QAOA mappings of various optimization problems with hard constraints, with the important property of restricting the quantum evolution to the subspace of feasible states [28, 29]. In particular, several mixing operators  $\Lambda_f(e^{-iH\alpha})$  proposed in [28] implement evolution under a local mixing Hamiltonian  $B$  controlled by a Boolean function  $f$ , where the control function checks that the mixing action on a given basis state will maintain feasibility and acts non-trivially only when this is the case. For example, for MaxIndependentSet on a given graph, the transverse-field (bit-flip) mixer is applied for each vertex controlled by the variables corresponding to the its neighbors in the graph; see [28] for details.

We next consider the special case where the target Hamiltonian  $H$  corresponds to a bit flip such that each computational basis state  $|y\rangle|0\rangle$  is mapped to  $|y\rangle|f(y)\rangle$ .

**1.1.5 Computing Boolean Functions in a Register.** We show how our results may be applied to construct explicit unitary operators which reversibly compute function values in an ancilla qubit register (i.e., implement *oracle queries*), as well as their corresponding Hamiltonians. We remark that a related approach for implementing such operators in a specific gate set has recently appeared [53]. Recall that in the computational basis, the Pauli  $X$  operator acts as  $X|0\rangle = |1\rangle$  and  $X|1\rangle = |0\rangle$ , i.e., as the *bit-flip*, or NOT, operation.

**PROPOSITION 2.** *For an  $n$ -bit Boolean function  $f$  represented by a Hamiltonian  $H_f$ , let  $G_f$  be the unitary self-adjoint operator on  $n + 1$  qubits which acts on computational basis states  $|x\rangle|a\rangle$  as*

$$G_f|x\rangle|a\rangle = |x\rangle|a \oplus f(x)\rangle, \quad (10)$$

where  $x \in \{0, 1\}^n$  and  $a \in \{0, 1\}$ . Then

$$G_f = \Lambda_f(X) = e^{-i\frac{\pi}{2} H_f \otimes (X-I)}, \quad (11)$$

and  $G_f = I$  if and only if  $f$  is unsatisfiable.

The proof is given in Section 4.2. As before it remains computationally hard to compute the Fourier expansion of such operators in general.

**COROLLARY 2.** *If  $f$  is given as a SAT formula in conjunctive normal form, then it is #P-hard to compute the identity coefficient  $\widehat{g}(\emptyset) = \text{tr}(G_f)/2^{n+1}$  of  $G_f$ , and NP-hard to decide if  $\widehat{g}(\emptyset) \neq 1$ .*

Equation (11) shows how Hamiltonian simulation may be used to compute a function  $f$  in a register. In particular, as the Pauli terms in  $H_f \otimes X$  and  $H_f \otimes I$  mutually commute,  $G_f$  can be implemented with  $2 \cdot \text{size}(H_f)$  many multi-qubit Pauli rotations with locality up to  $\deg(H_f) + 1$ .

## 2 REPRESENTING $n$ -BIT FUNCTIONS AS DIAGONAL QUBIT HAMILTONIANS

Many important problems, algorithms, and operators in physics and computer science naturally involve Boolean predicates. Indeed, the relationship between logical propositions and physical observables is foundational in quantum mechanics [11]. For the case of qubits, we show how the natural unique representation of a Boolean function as a Hamiltonian composed of spin operators (Pauli  $Z$  matrices) follows from classical Fourier analysis; see [20, 48] for overviews of the subject. We use these tools to extend our results to Hamiltonians representing more general functions built from sums, conjunctions, disjunctions, and other basic combinations of simpler Boolean clauses.

### 2.1 Boolean Functions

The class of Boolean functions on  $n$  bits is defined as  $\mathcal{B}_n := \{f : \{0, 1\}^n \rightarrow \{0, 1\}\}$ . Taken as the elements of a real vector space, for each  $n$  they give a basis for the real functions  $\mathcal{R}_n = \{f : \{0, 1\}^n \rightarrow \mathbb{R}\}$  on  $n$  bits. Moreover,  $\mathcal{R}_n$  is isomorphic to the vector space of diagonal Hamiltonians acting on  $n$  qubits, or, equivalently, the space of  $2^n \times 2^n$  diagonal real matrices. Thus, diagonal Hamiltonians naturally encode large classes of functions.

We say a Hamiltonian *represents* a function  $f$  if in the computational basis it acts as the corresponding multiplication operator, i.e., it satisfies the  $2^n$  eigenvalue equations

$$\forall x \in \{0, 1\}^n \quad H_f|x\rangle = f(x)|x\rangle. \quad (12)$$

On  $n$  qubits, this condition specifies  $H_f$  uniquely (up to the choice of the computational basis). Equivalently, we may write  $H_f = \sum_x f(x)|x\rangle\langle x|$ , which for a Boolean function  $f \in \mathcal{B}_n$  becomes

$$H_f = \sum_{x:f(x)=1} |x\rangle\langle x|. \quad (13)$$



As Boolean functions satisfy  $f^2 = f$  we have  $H_f^2 = H_f$ , so  $H_f$  is a projector<sup>2</sup> of rank  $r = \#f := |\{x : f(x) = 1\}| = \sum_x f(x)$ . Hence, the Hamiltonian  $H_f$  for a Boolean function  $f$  is equivalent to the projector onto the subspace spanned by basis vectors  $|x\rangle$  such that  $f(x) = 1$ , and given an  $f$  such a projector may be constructed using our results below. Hence, determining if  $f$  is satisfiable is equivalent to determining if  $H_f$  is not identically 0, and determining  $H_f$  explicitly in the form of (13) is at least as hard as counting the number of satisfying assignments of  $f$ , or equivalently, computing  $r = \text{rank}(H_f)$ .

We consider the standard computational basis of eigenstates of Pauli  $Z$  operators (often written as  $\sigma_z$ ), defined by the relations  $Z|0\rangle = |0\rangle$  and  $Z|1\rangle = -|1\rangle$ . We use  $Z_j = I \otimes \dots \otimes I \otimes Z \otimes I \dots \otimes I$  to denote  $Z$  acting on the  $j$ th qubit. Products of  $Z_j$  over a set of qubits act as

$$\prod_{j \in S} Z_j |x\rangle = \chi_S(x) |x\rangle, \quad (14)$$

where each *parity function*  $\chi_S(x) : \{0, 1\}^n \rightarrow \{-1, +1\}$  gives the parity of the bits of  $x$  in the subset  $S \subset [n]$ , i.e., is +1 if and only if the number of bits of  $x$  set to 1 is even. Identifying each  $S$  with its characteristic vector  $S \in \{0, 1\}^n$  such that  $S \cdot x = \sum_{j \in S} x_j$ , we have

$$\chi_S(x) = (-1)^{S \cdot x} = (-1)^{\oplus_{j \in S} x_j}. \quad (15)$$

Thus, each Hamiltonian  $Z_S := \prod_{j \in S} Z_j$  represents the function  $\chi_S(x)$  in the sense of (12).

The set of parity functions on  $n$  bits  $\{\chi_S(x) : S \subset [n]\}$  also gives a basis for the real functions  $\mathcal{R}_n$ . This basis is orthonormal with respect to the inner product defined by

$$\langle f, g \rangle := \frac{1}{2^n} \sum_{x \in \{0, 1\}^n} f(x)g(x). \quad (16)$$

Hence, every Boolean function  $f \in \mathcal{B}_n$  may be written uniquely as

$$f(x) = \sum_{S \subset [n]} \widehat{f}(S) \chi_S(x), \quad (17)$$

called the *Fourier expansion* (or, sometimes, *Walsh* or *Hadamard* expansion [41], or *phase polynomial* [2]) with *Fourier coefficients* given by the inner products of  $f$  with the parity functions

$$\widehat{f}(S) = \frac{1}{2^n} \sum_{x \in \{0, 1\}^n} f(x) \chi_S(x) = \langle f, \chi_S \rangle, \quad (18)$$

and satisfying *Parseval's identity*

$$\sum_{S \subset [n]} \widehat{f}(S)^2 = \frac{1}{2^n} \sum_x f(x)^2. \quad (19)$$

The quantity  $\sum_{S \neq \emptyset} \widehat{f}(S)^2 =: \text{var}(f)$  is often referred to as the *variance* of  $f$ . For  $\{0, 1\}$ -valued functions, we have  $f^2 = f$  which implies  $\sum_{S \subset [n]} \widehat{f}(S)^2 = \frac{1}{2^n} \sum_x f(x) = \widehat{f}(\emptyset) = \widehat{f}(\emptyset)^2 + \text{var}(f)$ .

We refer to the mapping from  $f(x)$  to  $\widehat{f}(S)$  as the *Fourier transform* of  $f$ . The sparsity of  $f$ , denoted  $\text{spar}(f)$ , is the number of non-zero coefficients  $\widehat{f}(S)$ . When  $\text{spar}(f) = \text{poly}(n)$  we refer to  $f$  as *polynomially sparse*; in particular, polynomially sparse functions yield Hamiltonians of size  $\text{poly}(n)$ . The *degree* of  $f$ , denoted  $\deg(f)$ , is defined to be the largest  $|S|$  such that  $\widehat{f}(S)$  is non-zero. Note that if  $f$  depends on only  $k \leq n$  variables, then  $\deg(f) \leq k$ .

<sup>2</sup>Projectors give quantum observables. In particular, for an arbitrary normalized  $n$ -qubit state  $|\psi\rangle$ , the probability  $p_1$  of a computational basis measurement returning a satisfying bit string (i.e., an  $x$  such that  $f(x) = 1$ ) is given by  $p_1 = \langle \psi | H_f | \psi \rangle$ , i.e., is equal to the expected value of repeated measurements of  $H_f$  on the state  $|\psi\rangle$ .

We summarize our results on the representation of Boolean functions in Theorem 1 above.

**PROOF OF THEOREM 1.** From the identification of  $\chi_S$  with  $Z_S = \bigotimes_{j \in S} Z_j = \prod_{j \in S} Z_j$ , from (17) we see that each Boolean function  $f$  is represented as a diagonal Hamiltonian by a linear combination of tensor products of  $Z_j$  operators. As all Pauli operators and their tensor products are traceless except for the identity operator, we have  $\widehat{f}(S) = \text{tr}(H_f \prod_{j \in S} Z_j)/2^n$  where<sup>3</sup> the results (4) and (5) follow from Parseval's identity (19) using  $f^2 = f$ . Recall we define the degree (sometimes called the *Pauli weight*) of such a Hamiltonian  $H_f$ ,  $\deg(H_f)$ , to be the largest number of qubits acted on by any term in this sum, and the size,  $\text{size}(H_f)$ , to be the number of (non-zero) terms. Clearly, we have  $\deg(H_f) = \deg(f) =: d$ , and simple counting and the bound  $\binom{n}{d} \leq n^d (e/d)^d / e$  gives  $\text{size}(H_f) = \text{spar}(f) \leq d \binom{n}{d} \leq (e/d)^{d-1} n^d + 1$ .  $\square$

We emphasize that the Hamiltonian coefficients  $\widehat{f}(S)$  depend only on the function values  $f(x)$ , and are independent of *how* such a function may be represented as input (e.g., formula, circuit, and truth table). Many typical compact representations of Boolean functions as computational input, such as Boolean formulas or Boolean circuits can be directly transformed to Hamiltonians using the composition rules of Theorem 2 which we derive below.

*Remark 3.* The identification of Boolean functions as real polynomials allows application of Theorem 1 to different domains or targets. Changing the target from  $\{0, 1\}$  to  $\{1, -1\}$  for a given  $f(x)$  corresponds to the function  $g(x) = 2f(x) - 1$  with Hamiltonian coefficients  $\widehat{g}(\emptyset) = 1 - 2\widehat{f}(\emptyset)$  and  $\widehat{g}(S) = -2\widehat{f}(S)$  for  $S \neq \emptyset$ , so this can change  $\text{size}(H_f)$  by at most 1. (In this case there are several differences from the  $\{0, 1\}$  approach; for example, Parseval's identity trivially becomes  $\sum_{S \subseteq [n]} \widehat{f}(S)^2 = 1$ .)

Similarly, indeed, the Fourier expansion (17) itself corresponds to changing the domain from  $\{0, 1\}^n$  to  $\{1, -1\}^n$  (i.e., the polynomial obtained replacing each  $Z_j$  in (6) by the variable  $z_j = (-1)^{x_i} \in \{-1, 1\}$ ).

On the other hand, Theorem 1 shows that computing the Hamiltonian representation of a Boolean function is as hard as computing its number of satisfying assignments, which is believed to be a computationally intractable problem in general [3]. We illustrate this explicitly in Corollary 1 above where we show computing  $\widehat{f}(\emptyset)$  can be #P-hard. Moreover, arbitrary Boolean functions may have size (sparsity) exponential in  $n$ , in which case, even if we know somehow its Hamiltonian representation, we cannot implement or simulate this Hamiltonian efficiently (with respect to  $n$ ) with the usual direct approaches.

As explained, Hamiltonians representing pseudo-Boolean functions often avoid these difficulties; for example, constraint satisfaction problems with objective function given as the sum of a number of *local* clauses each clause acting on at most  $k = O(\log n)$  bits (e.g., Max- $k$ -Sat), and so the Hamiltonians representing each clause have degree  $O(\log n)$  and size  $O(\text{poly}(n))$ , and hence can be efficiently constructed. Applying the well-known results of [46, Thm. 1 and 2] for such functions to Theorem 1 immediately gives the following useful Hamiltonian degree bounds.

**COROLLARY 3.** *For a function  $f \in \mathcal{B}_n$  that depends only on  $k \leq n$  variables, represented as a Hamiltonian  $H_f$  acting on  $n$  qubits, the degree of  $H_f$  satisfies*

$$k \geq D(f) \geq \deg(H_f) \geq \log_2 k - O(\log \log k), \quad (20)$$

where  $D(f)$  is the decision tree complexity of  $f$  and  $D(f) = O(\text{poly}(\deg(H_f)))$ .

<sup>3</sup> $\text{tr}(H)$  denotes the *trace* of the matrix  $H$ , i.e., the (basis-independent) sum of its diagonal elements, .

We emphasize that further results from the literature concerning the Fourier analysis of Boolean functions may be similarly adapted to obtain properties of corresponding Hamiltonians. In particular additional useful details of the Fourier coefficients may be found in [2, 20, 48].

## 2.2 Basic Clauses and Logical Composition Rules

Boolean functions arise in many different applications, but are often given in or easily reduced to a *normal form*. For example, SAT formulas are given in conjunctive normal form. Many other normal forms exist such as disjunctive, algebraic ( $\oplus$ ), minterm or maxterm, and so on [26]. Note that while logically equivalent, the different forms may be quite different for computational purposes. For each form there corresponds a notion of size (which directly relates to the number of bits needed to describe a function in such a form).

The laws of classical propositional logic allow for convenient manipulation of a given Boolean expression between logically equivalent forms. Hence, given Hamiltonians representing basic Boolean functions, it is useful to have a methodical way to combine them too in order to construct new Hamiltonians representing their logical conjunctions (AND), disjunctions (OR), and so on. This approach often allows for easier construction of such a Hamiltonian than by working with the Fourier expansion directly. First, consider Hamiltonians representing basic functions and variables. The trivial functions  $f = 1$  and  $f = 0$  (respectively, *true* and *false*) are represented by the  $2^n$ -dimensional Hamiltonians  $H_1 = I$  and  $H_0 = 0$ , respectively. Using the identity  $(-1)^x = 1 - 2x$ , we represent the  $j$ th variable  $x_j$ , considered as a multiplication operator, as the Hamiltonian

$$H_{x_j} = I^{\otimes j-1} \otimes |1\rangle\langle 1| \otimes I^{\otimes n-j} = \frac{1}{2}I - \frac{1}{2}Z_j, \quad (21)$$

which acts according to the value of the  $j$ th bit as  $H_{x_j}|x\rangle = x_j|x\rangle$ . Similarly, the logical negation of the  $j$ th variable is represented as  $H_{\bar{x}_j} = I/2 + Z_j/2$ . For clarity we will avoid writing tensor factors of identity operators explicitly when there is no ambiguity, and for convenience we sometimes write  $x_j$  to mean the Hamiltonian  $H_{x_j}$ , and likewise for other basic functions such as  $\bar{x}_j$ .

Applying the laws of propositional logic leads to the composition rules of Theorem 2.

**PROOF OF THM. 2.** The logical values 1 and 0 (i.e., *true* and *false*) are represented as the identity matrix  $I$  and the zero matrix, respectively. Each result follows from the natural embedding of  $f, g \in \mathcal{B}_n$  into  $\mathcal{R}_n$ , the real vector space of real functions on  $n$  bits. From linearity of the Fourier transform, we immediately have  $H_{af+bg} = aH_f + bH_g$  for  $a, b \in \mathbb{R}$ . Using standard identities, the Boolean operations  $(\cdot, \vee, \oplus, \dots)$  on  $f, g$  can be translated into  $(\cdot, +)$  formulas, i.e., linear combinations of  $f$  and  $g$ . Linearity then gives the resulting Hamiltonian in terms of  $H_f$  and  $H_g$ . Explicitly, for the complement of a function  $\bar{f}$ , as  $\bar{f} = 1 - f$ , we have  $H_{\bar{f}} = I - H_f$ . Similarly, the logical identities  $f \wedge g = fg$ ,  $f \vee g = f + g - fg$ ,  $f \oplus g = f + g - 2fg$ , and  $f \Rightarrow g = \bar{f} + fg$ , respectively, imply the remaining results of the theorem.  $\square$

We summarize the Hamiltonian representations of several basic Boolean functions in Table 1 above, which are easily derived from Theorem 1. Applying the laws of Theorem 2 we may derive Hamiltonians representing more complicated Boolean formulas than those of Table 1, such as expressions with arbitrary numbers of variables, mixed types of clauses, or given as Boolean circuits. Some typical examples of Boolean functions on 3 variables are the Majority (MAJ), Not-All-Equal (NAE), and 1-in-3 functions, which behave as their names indicate. The  $\text{Mod}_3$  function is 1 when the sum  $x_1 + x_2 + x_3$  is divisible by 3, and satisfies  $\text{Mod}_3 = \text{NAE}$ . We show the Hamiltonians representing these functions in Table 2, which may be derived using either the composition rules of Theorem 2 or the Fourier expansion approach of Theorem 1. For example,  $H_{1in3}$  follows applying Thm. 2 with the identity  $1in3(x_1, x_2, x_3) = x_1\bar{x}_2\bar{x}_3 + \bar{x}_1x_2\bar{x}_3 + \bar{x}_1\bar{x}_2x_3$ .

The rules of Theorem 2 may be applied *recursively* to construct Hamiltonians representing more complicated Boolean functions, corresponding e.g., to parentheses in logical formulas, or wires in Boolean circuits. For example, the Hamiltonian representing the Boolean clause  $f \vee g \vee h = f \vee (g \vee h)$  is given by  $H_{f \vee g \vee h} = H_f + H_{g \vee h} - H_f H_{g \vee h}$ , which simplifies to

$$H_{f \vee g \vee h} = H_f + H_g + H_h - H_f H_g - H_f H_h - H_g H_h + H_f H_g H_h.$$

Another example is the Majority function which satisfies

$$H_{MAJ(f,g,h)} = -2H_f H_g H_h + H_f H_g + H_f H_h + H_g H_h.$$

Together, the rules of Theorem 2 and Table 1 show how to construct Hamiltonians representing functions given as arbitrary Boolean algebra ( $\wedge, \vee$ ) or Boolean ring ( $\cdot, \oplus$ ) elements, which are functionally complete in the sense of representing all possible Boolean functions [26].

### 2.3 Pseudo-Boolean Functions and Constraint Satisfaction Problems

Real functions on  $n$  bits are similarly represented as diagonal Hamiltonians via their Fourier expansion. Every such function  $f \in \mathcal{R}_n$  may be expanded (non-uniquely) as a weighted sum of Boolean functions, possibly of exponential size. By linearity of the Fourier transform, the Hamiltonian  $H_f$  is given precisely by the corresponding weighted sum of the Hamiltonians representing the Boolean functions. Moreover, the Hamiltonian  $H_f$  is unique, so different expansions of  $f$  as sums of Boolean functions must all result in the same  $H_f$ .

The Fourier coefficients are again given by the inner product (16) with the parity functions  $\chi_S$ ,

$$\widehat{f}(S) = \langle f, \chi_S \rangle = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x) \chi_S(x), \quad (22)$$

and satisfy Parseval's identity as stated in (19). We are particularly interested in *pseudo-Boolean functions* given as a weighted sum of logical clauses

$$f(x) = \sum_{j=1}^m w_j f_j(x), \quad (23)$$

where  $f_j \in \mathcal{B}_n$  and  $w_j \in \mathbb{R}$ . Note that we do not deal explicitly with how the real numbers  $w_j$  are represented and stored; for many applications they are bounded rational numbers and this issue is relatively minor; see, e.g., the constructions in [28, 40]. Indeed, in a constraint satisfaction problem, typically all  $w_j = 1$  and hence  $f(x)$  gives the number of satisfied clauses (constraints).

We have the following result which extends the previous results for Boolean functions.

**PROOF OF THEOREM 3.** By the linearity of the Fourier expansion and Theorem 1 we have that an  $n$ -bit real function  $f : \{0,1\}^n \rightarrow \mathbb{R}$  is represented as the Hamiltonian  $H_f = \sum_{S \subseteq [n]} \widehat{f}(S) \prod_{j \in S} Z_j$  with  $\widehat{f}(S) = \langle f, \chi_S \rangle = \frac{1}{2^n} \text{tr}(H_f \prod_{j \in S} Z_j) \in \mathbb{R}$ . For pseudo-Boolean functions  $f = \sum_{j=1}^m w_j f_j$  the bounds  $d := \deg(H_f) \leq \max_j \deg(f_j)$  and  $\text{size}(H_f) \leq \min\{\sum_j \text{size}(H_{f_j}), (e/d)^{d-1} n^d + 1\}$  follow from simple counting as in the proof of Theorem 1.  $\square$

Thus, the results of Theorems 1 and 2 for Boolean functions also apply to the construction of Hamiltonians representing real functions, though with several important distinctions. Various subclasses of so-called pseudo-Boolean functions with particular attributes (e.g., submodularity) are important in applications and their properties may be further studied; see [13] for an overview.

*Remark 4.* In contrast to Theorem 1, for a constraint satisfaction problem  $f = \sum_{j=1}^m f_j$ , with  $f_j \in \mathcal{B}_n$ , applying Parseval's identity (19) we have

$$\sum_{S \subseteq [n]} \widehat{f}(S)^2 = \mathbb{E}[f] + 2 \sum_{i < j} \langle f_i, f_j \rangle = \widehat{f}(\emptyset) + 2 \sum_{i < j} \mathbb{E}[f_i \wedge f_j] \geq \mathbb{E}[f],$$

where  $\mathbb{E}[f] := \frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x)$  gives the expected value of  $f(x)$  over the uniform distribution. In particular,  $\sum_{S \subseteq [n]} \widehat{f}(S)^2 = \mathbb{E}[f]$  if and only if  $\langle f_i, f_j \rangle = 0$  for all  $i, j$ . If there does exist an  $i, j$  such that  $\langle f_i, f_j \rangle = 0$ , then the conjunction of the clauses is unsatisfiable, i.e.,  $\bigwedge_j f_j = 0$ .

### 3 APPLICATIONS TO CONSTRUCTING DIAGONAL OPERATORS

Here we overview four immediate applications of our results to diagonal operators and problem encodings: constrained optimization, and ground state logic, simulating diagonal Hamiltonians, and quadratic optimization.

#### 3.1 Constrained Optimization

In constrained optimization problems, we seek an optimal solution subject to satisfying a set of *feasibility* constraints. These constraints may arise as part of the problem itself or from its encoding [28, 40]. In quantum annealing, a common approach to dealing with problem constraints is to augment the Hamiltonian that represents the objective function to be minimized with additional diagonal Hamiltonian terms that penalize (i.e., shift the eigenvalues of) states outside of the feasible subspace [40, 42].

Suppose we are given a constrained real function  $f(x)$  to minimize, with a set of Boolean hard constraint functions  $g_j$ ,  $j = 1, \dots, \ell$ , such that at least one of the  $g_j(x) = 1$  if  $x$  is an infeasible solution, and so  $\sum_j g_j(x) = 0$  if and only if  $x$  is feasible. (Here, we may always redefine  $f$  as to take a convenient value on infeasible states.) We may construct the augmented problem Hamiltonian as

$$H_p = H_f + \sum_{j=1}^{\ell} w_j H_{g_j},$$

where the Hamiltonians  $H_f$  and  $H_{g_j}$  represent  $f$  and the  $g_j$  as in Theorems 3 and 1, respectively. The  $w_j$  are positive weights which may be selected appropriately such that infeasible basis states are eigenvectors of  $H_p$  with eigenvalues shifted away from those of  $f$  (e.g.,  $w_j > \max(x) f(x)$ ), and feasible states are eigenvectors with eigenvalues  $f(x)$ . Hence, the ground state subspace of  $H_p$  is spanned by states representing optimal feasible problem solutions. Theorems 2, 1, and 3 may be applied to the functions  $g_j$  to construct the penalty terms  $H_{g_j}$  just as for the cost term  $H_f$ .

Thus, our results may also be applied to explicitly construct problem mappings with penalty terms for constrained problems. See [40] for a number of specific problem mappings; we emphasize our approach may be applied directly to problems not considered therein. Furthermore, similar ideas apply to related approaches for constrained optimization such as Lagrange multipliers [49].

An alternative approach to constrained optimization is to map the hard constraint functions to diagonal Hamiltonians, which may be used to design mixing Hamiltonian that preserves the subspace of feasible states. This approach is proposed in constraint-preserving generalizations of quantum annealing [31, 32] and QAOA [28, 29], and can offer advantages over penalty-term approaches; see [28, 29, 31, 32] for details and example problem mappings.

#### 3.2 Ground State Boolean Logic

With a universal quantum gate-model computer, a single (additional) control bit suffices for all efficiently computable control functions [44]. Indeed, in principle we can always compute a Boolean

function  $f(x)$  in a control register by constructing a unitary operator  $U_f : |x\rangle|a\rangle \rightarrow |x\rangle|a \oplus f(x)\rangle$ , as we consider in Section 4.2. Nevertheless, in certain models or applications it is desirable to have a purely Hamiltonian implementation.

A different approach to computing a Boolean function  $f \in \mathcal{B}_n$  in a register is to encode its input-output pairs as the ground state subspace of a Hamiltonian  $H$ , referred to as *ground state Boolean logic*; see, e.g., [9, 18, 57]. There are different ways to encode a function as the ground state subspace, as in there is freedom in how the Hamiltonian acts on invalid computational basis states. For example, the function  $AND(x, y) = xy$  can be encoded with the subspace  $\text{span}\{|x\rangle|y\rangle|xy\rangle\} = \text{span}\{|000\rangle, |010\rangle, |100\rangle, |111\rangle\}$ , which corresponds to the ground state subspace of a number of Hamiltonians. To construct such a Hamiltonian, the penalty term approach of the previous section could be used to penalize the invalid states. An alternative construction that takes advantage of the Hamiltonian representation  $H_f$  is to directly implement the Boolean function  $g \in \mathcal{B}_{n+1}$  that satisfies  $g(x, y) = 0$  if and only if  $y = f(x)$ , or equivalently  $g(x, y) = \overline{f(x) \oplus y}$ . Applying Theorems 1 and 2 to this function and simplifying gives the following result.

**PROPOSITION 3.** *Let  $f \in \mathcal{B}_n$  be represented by the Hamiltonian  $H_f$  as in Theorem 1, and  $x_a := \frac{1}{2}I - \frac{1}{2}Z_a$  where the ancilla qubit is labeled  $a$ . Then the  $(n + 1)$ -qubit Hamiltonian  $H_g$*

$$H_g = I \otimes x_a + H_f \otimes Z_a, \quad (24)$$

*represents the Boolean function  $g \in \mathcal{B}_{n+1}$  which satisfies  $g(x, y) = 0$  if and only if  $y = f(x)$ , and has ground state subspace given by*

$$\text{span}\{|x\rangle|f(x)\rangle : x \in \{0, 1\}^n\}.$$

Simpler Hamiltonians with the same ground state subspace may be found for specific classes of Boolean functions  $f$ ; see, e.g., [10]. An advantage of the construction (24) is that it applies generally.

### 3.3 Simulating Diagonal Hamiltonians

Here, we explain how Theorems 1 to 3 may be straightforwardly applied to yield quantum circuits implementing time evolution under diagonal Hamiltonians, i.e., diagonal unitaries, which is a fairly well-studied problem with several related approaches proposed in the literature.

In many applications we desire to *simulate* a Hamiltonian  $H$  for some time  $\gamma \in \mathbb{R}$ , i.e., implement exactly or approximately the unitary operator  $U(\gamma) = e^{-i\gamma H}$ . When  $H$  is diagonal, efficient quantum circuits may be obtained using Fourier analysis [2, 51, 56] or other approaches. Consider the simulation of a Hamiltonian  $H_f$  representing a real or Boolean function  $f$ . It is well known that if  $f$  can be efficiently computed classically, and if ancilla qubits are available, then the Hamiltonian  $H_f$  can be simulated efficiently by computing  $f$  in a scratchpad register and performing a sequence of controlled rotations; see, e.g., [17]. These methods typically avoid computing the Fourier expansion of  $f$  explicitly. On the other hand, there exist applications where an explicit Hamiltonian-based implementation is desirable, such as quantum annealing, or cases where we wish to minimize the need for ancilla qubits, such as, for example, near-term implementations. Efficient circuits simulating products of Pauli  $Z$  operators are well-known [6, 17], as shown in Figure 1. As Pauli  $Z$  terms mutually commute, circuits simulating individual terms in the Hamiltonians (2) or (6) can be applied in any sequence as to simulate their sum. Thus, when  $\text{size}(H_f) = O(\text{poly}(n))$  we can always simulate  $H_f$  efficiently in this way. We summarize this observation in the following.

**COROLLARY 4.** *A Hamiltonian  $H_f$  representing a Boolean or real function  $f$  as in (2) or (6) can be simulated, i.e., the operation  $\exp(-i\gamma H_f)$  implemented, with  $n$  qubits and  $O(\deg(H_f) \cdot \text{size}(H_f))$  basic quantum gates. In particular, Hamiltonians  $H_f$  with bounded maximum degree  $d := \deg(H_f) = O(1)$  can be simulated with  $O(n^d)$  basic gates. Ancilla qubits are not necessary in either case.*



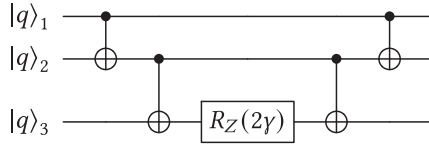


Fig. 1. Quantum circuit performing the operation  $U = \exp(-i\gamma Z_1 Z_2 Z_3)$  on three qubits labeled 1, 2, and 3. The middle operator is a Z-rotation gate, and the other gates are controlled-NOT (CNOT) gates with a black circle indicating the control qubit and cross indicating the target. By similar circuits,  $U = \exp(-i\gamma Z_1 Z_2 \dots Z_\ell)$  can be implemented with  $2(\ell - 1)$  CNOT gates and one  $R_Z$  gate. Different circuit compilations are possible, including compilation to different gate sets.

Here, by *basic quantum gates* we mean the set of CNOT and single qubit rotation gates, which is a standard universal set [6, 44]. We remark that the Hamiltonian simulation considered in the corollary is exact in the sense that if each of the basic gates is implemented exactly, then so is  $\exp(-i\gamma H_f)$ . The approximation of quantum gates and operators is an important topic but we do not deal with it here; see, e.g., [44].

*Example 1 (Application to Grover's Algorithm).* For a Boolean function  $f$ , simulating  $H_f$  for time  $\pi$  gives the standard oracle query for Grover's algorithm [44]

$$e^{-i\pi H_f} |x\rangle = (-1)^{f(x)} |x\rangle. \quad (25)$$

Hence, when  $H_f$  is known explicitly and  $\text{size}(H_f) = \text{poly}(n)$ , we can efficiently construct and implement the operator  $(-1)^{f(x)}$  using quantum circuits for simulating  $H_f$  using only CNOT and  $R_Z$  gates, without any necessary ancilla qubits.

### 3.4 Quadratic Unconstrained Binary Optimization

A general and important class of pseudo-Boolean optimization problems are **quadratic unconstrained binary optimization (QUBO)** problems [42], where we seek to maximize or minimize a degree-two pseudo-Boolean function

$$f(x) = a + \sum_{j=1}^n c_j x_j + \sum_{j < k} d_{jk} x_j x_k, \quad (26)$$

with  $a, c_j, d_{jk} \in \mathbb{R}$ , and  $x_j \in \{0, 1\}$ . Indeed, this is the class of problems (ideally) implementable on current quantum annealing devices such as, for example, D-WAVE machines, where the qubit interactions are themselves quadratic [10, 42]. The QUBO class also contains many problems which at first sight are not quadratic, via polynomial reductions which often require extra variables; indeed, the natural QUBO decision problem is NP-complete [24]. Note that  $\bar{x}_j = 1 - x_j$ , so (26) is without loss of generality. Applying our above results gives the following.

**COROLLARY 5.** *The QUBO objective function (26) maps to a Hamiltonian given as a quadratic sum of Pauli Z operators, with  $\text{size}(H_f) \leq 1 + n/2 + n^2/2$ . Explicitly, we have*

$$H_f = (a + c + d)I - \frac{1}{2} \sum_{j=1}^n (c_j + d_j) Z_j + \frac{1}{4} \sum_{j < k} d_{jk} Z_j Z_k, \quad (27)$$

where we have defined  $c = \frac{1}{2} \sum_{j=1}^n c_j$ ,  $d = \frac{1}{4} \sum_{j < k} d_{jk}$ , and  $d_j = \frac{1}{2} \sum_{k: k \neq j} d_{jk}$  with  $d_{jk} = d_{kj}$ .

Moreover, we can simulate  $H_f$ , i.e., implement the phase operator  $U_p(t) = e^{-itH_f}$ , using at most  $n$  many  $R_Z$  rotation gates and  $\binom{n}{2}$  many  $R_{ZZ}$  gates.

The QUBO problem is closely related to the Ising model of interacting spins from physics [47].

*Example 2 (Interacting Ising Spins).* Consider the related classical ISING decision problem of determining whether the ground state energy (lowest eigenvalue) of an Ising model (degree two) diagonal Hamiltonian

$$H = a_0 I + \sum_j a_j Z_j + \sum_{j < k} a_{jk} Z_j Z_k$$

is at most a given constant. As the number of terms is  $\text{size}(H) = O(n^2)$ , we can efficiently check the energy  $H(x)$  of each candidate ground state  $|x\rangle$ ,  $x \in \{0, 1\}^n$ , so the problem is in NP. On the other hand, from Theorem 2, we see that the NP-complete problem MAX-2-SAT maps to a degree-two Hamiltonian of this form, with solution encoded in the ground state energy of  $-H$ . Thus, from our results it trivially follows that ISING is NP-complete. Similar arguments can be used to show NP-completeness with restricted coefficients values (e.g., antiferromagnetic) or with restricted interaction topologies such as planar graphs [5].

#### 4 APPLICATIONS TO NON-DIAGONAL OPERATORS

Here we consider the application of our results beyond strictly diagonal operators, in particular to constructing controlled unitaries (quantum gates) common in quantum algorithms.

Just as we have seen that diagonal Hamiltonians correspond to linear combinations of Pauli Z operators, tensor products of arbitrary single-qubit Pauli matrices give a basis for the vector space of  $n$ -qubit Hamiltonians. Indeed, a general Hamiltonian  $H$  may be expanded as a sum of Pauli matrices as

$$H = a_0 I + \sum_{j=1}^n \sum_{\sigma=X,Y,Z} a_{j\sigma} \sigma_j + \sum_{j \neq k} \sum_{\sigma=X,Y,Z} \sum_{\lambda=X,Y,Z} a_{jk\sigma\lambda} \sigma_j \lambda_k + \dots, \quad (28)$$

with real coefficients  $a_\alpha \in \mathbb{R}$ . (For general linear operators on qubits the same formula holds but with  $a_\alpha \in \mathbb{C}$ , see e.g., [58].) The coefficients are easily shown to satisfy

$$a_\alpha = \frac{1}{2^n} \text{tr}(\alpha H), \quad (29)$$

for each of the  $4^n$  Pauli terms  $\alpha = \sigma_1 \sigma_2 \dots \sigma_n$ ,  $\sigma_j \in \{I, X_j, Y_j, Z_j\}$ , which are orthonormal with respect to the Hilbert-Schmidt inner product  $\langle \alpha, \beta \rangle := \frac{1}{2^n} \text{tr}(\alpha^\dagger \beta)$  that generalizes (16).

##### 4.1 Controlled Unitaries and Hamiltonians

Many quantum algorithms require controlled Hamiltonian evolutions. For example, in **quantum phase estimation (QPE)** [44], we require transformations on  $(1+n)$ -qubit basis states of the form

$$|0\rangle|x\rangle \rightarrow |0\rangle|x\rangle, \quad |1\rangle|x\rangle \rightarrow |1\rangle e^{-iHt} |x\rangle,$$

for various values  $t = 1, 2, 4, \dots$ . Consider such a transformation with fixed  $t$ . Labeling the first register (control qubit)  $a$ , the overall unitary may be written as

$$\Lambda_{x_a}(e^{-iHt}) = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes e^{-iHt}. \quad (30)$$

Recall the notation  $\Lambda_{x_a}(e^{-iHt})$  indicates the unitary  $e^{-iHt}$  controlled by the classical function  $x_a$ . We obtain the Hamiltonian corresponding to this transformation by writing  $\Lambda_{x_a}(e^{-iHt}) = e^{-i\tilde{H}t}$ , which gives

$$\tilde{H} = |1\rangle\langle 1| \otimes H = x_a \otimes H = \frac{1}{2} I \otimes H - \frac{1}{2} Z_a \otimes H. \quad (31)$$

Recall that for simplicity we sometimes write a function  $f$  in place of its Hamiltonian representation  $H_f$  as we have done here for the function  $f(x) = x_a$ . Note that the control qubit is assumed precomputed here; its value may or may not depend on  $x$ .

More generally, consider Hamiltonian evolution controlled by a Boolean function  $g \in \mathcal{B}_k$  acting on a  $k$ -qubit ancilla register. In this case, we seek to affect the unitary transformation on  $(k + n)$ -qubit basis states

$$\begin{aligned} |y\rangle|x\rangle &\rightarrow |y\rangle|x\rangle && \text{if } g(y) = 0, \\ |y\rangle|x\rangle &\rightarrow |y\rangle e^{-iHt} |x\rangle && \text{if } g(y) = 1, \end{aligned}$$

which gives the overall unitary

$$\Lambda_g(e^{-iHt}) = \sum_{y:g(y)=0} |y\rangle\langle y| \otimes I + \sum_{y:g(y)=1} |y\rangle\langle y| \otimes e^{-iHt} = H_{\bar{g}} \otimes I + H_g \otimes e^{-iHt}, \quad (32)$$

corresponding to evolution under the Hamiltonian

$$\tilde{H}_g = \sum_{y:g(y)=1} |y\rangle\langle y| \otimes H = H_g \otimes H. \quad (33)$$

These results have been summarized in Proposition 1 above. Note that if the Hamiltonian  $H = H_f$  represents a Boolean function  $f$ , then (33) represents the conjunction of  $f$  and  $g$ , i.e., we have  $\tilde{H}_g = H_g \otimes H_f = H_{g \wedge f}$ .

Moreover, for an arbitrary Hamiltonian  $H$ , if we can implement the ancilla controlled operator  $\Lambda_{x_a}(e^{-iHt})$  for sufficiently many values of  $t \in \mathbb{R}$ , then it is straightforward to implement a variable-time controlled Hamiltonian simulation operator  $\Lambda_{\tau,H}$ , which acts on basis states as

$$\Lambda_{\tau,H}|\tau\rangle|x\rangle = |\tau\rangle e^{-iH\tau} |x\rangle. \quad (34)$$

For example, if  $\tau$  was encoded in binary, then  $\Lambda_H$  could be implemented by applying the operators  $\Lambda_{x_{a_0}}(e^{-iH2^0}), \Lambda_{x_{a_1}}(e^{-iH2^1}), \dots, \Lambda_{x_{a_j}}(e^{-iH2^j}), \dots$  in sequence controlled over each bit  $\tau_j$  in the first register; see, e.g., [44].

Finally, we remark on the condition that the control function and target unitaries act on disjoint sets of qubits, and the relation to spin creation and annihilation operators.

*Example 3 (Creation/annihilation Operators and Interacting Fermions).* Consider the operator  $Xx$  which acts as  $Xx|0\rangle = 0$  and  $Xx|1\rangle = |0\rangle$ . Clearly  $Xx = X(I - Z)/2 = X/2 + iY/2$  is not self-adjoint. Indeed,  $Xx$  is equivalently the well-known *spin annihilation* operator  $b = |0\rangle\langle 1|$ . The *spin creation* operator  $b^\dagger$  is similarly given as  $b^\dagger = (Xx)^\dagger = xX = X\bar{x} = |1\rangle\langle 0|$ , and the spin number (occupation) operator is  $b^\dagger b = x = (I - Z)/2$ . Thus, the product of two Hamiltonians  $H_g$  and  $H$  acting non-trivially on overlapping qubits is not guaranteed to yield a self-adjoint operator as in (33).

For systems of fermions in the second quantized (occupation number) representation [54], Hamiltonians consist of polynomials of *fermionic creation and annihilation operators*  $a^\dagger$  and  $a$ , satisfying the *canonical anticommutation relation* algebra  $\{a_j, a_k\} = \{a_j^\dagger, a_k^\dagger\} = 0$  and  $\{a_j, a_k^\dagger\} = \delta_{jk}$ . It is easily shown that these relations are satisfied if we use spin operators and the parity functions  $\chi_S$ ,  $S \subset [n]$  of (15) to represent the fermionic operators as  $a_j = \chi_{\{1,2,\dots,j-1\}} b_j$ . Hence, from above we may represent these operators as  $a_j = Z_1 Z_2 \dots Z_{j-1} (X_j + iY_j)/2$  and  $a_j^\dagger = Z_1 Z_2 \dots Z_{j-1} (X_j - iY_j)/2$  which reproduces the well-known *Jordan-Wigner transform*. This representation is used in many applications, such as quantum algorithms for quantum chemistry, though alternative mappings with different tradeoffs are known [52].

## 4.2 Computing Functions in Registers

Here we consider the special case of computing a Boolean function in a register and show a similar complexity result as for the diagonal Hamiltonian case. As an illustrative example we briefly consider the connection of our results to quantum query complexity and Grover's algorithm.

Suppose for a Boolean function  $f$  we have a unitary operator  $G_f$  that acts on each  $(n + 1)$ -qubit basis state  $|x\rangle|a\rangle$ ,  $a \in \{0, 1\}$  as

$$G_f|x\rangle|a\rangle = |x\rangle|a \oplus f(x)\rangle. \quad (35)$$

If we let the function  $f$  be arbitrary and unknown, then each application of  $G_f$  (considered a black-box) is called an *oracle query* for  $f$ . Operators  $G_f$  as in (35) are called *bit queries*. Note that  $G_f$  may often be derived from a reversible classical circuit for computing  $f$ , but we consider it abstractly here. (The query (25) for Grover's algorithm is often referred to as a *phase query*.)

We show that  $G_f$  also induces a diagonal representation of  $H_f$ , and hence faces the same computational difficulties for the Boolean case. Observe that the diagonal Hamiltonian

$$H'_f := G_f x_a G_f = \frac{1}{2}I - \frac{1}{2}G_f Z_a G_f = x_a + H_f Z_a \quad (36)$$

acts on computational basis states as

$$H'_f|x\rangle|a\rangle = (f(x) \oplus a)|x\rangle|a\rangle, \quad (37)$$

so we identify  $H'_f = H_{f \oplus a}$ . Hence when  $a = 0$ ,  $H'_f$  gives an  $n + 1$  qubit representation of  $f$ .

**PROOF OF PROP. 2.** The first statement follows from applying Prop. 1 with  $H_f$  as given from Thm. 1 and  $H = X_a$ . For the second statement, we use (28), (29), and (37) to expand  $G_f$  as a sum of Pauli terms acting on  $n$  qubits  $G_f = (1 - \widehat{f}(\emptyset))I + \widehat{f}(\emptyset)X_a + \dots$ , where none of the terms to the right are proportional to  $I$ . Hence the identity coefficient  $\widehat{g}(\emptyset) = \text{tr}(G_f)/2^{n+1} = 1 - \widehat{f}(\emptyset)$  gives (one minus) the fraction of satisfying assignments of  $f$ .  $\square$

We conclude the section with a pedagogic example application to oracle models.

*Example 4 (Phase Kickback and Function Queries).* Recall Example 1. For an arbitrary Boolean function  $f$ , a single application of the *bit-query oracle*  $G_f$  suffices to simulate  $(-1)^f$  as

$$G_f|x\rangle|-\rangle_a = (-1)^{f(x)}|x\rangle|-\rangle_a, \quad (38)$$

using a single ancilla qubit prepared in the state  $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = H|1\rangle$  and where  $H$  denotes the Hadamard (single qubit quantum Fourier transform) gate [44], known as phase kickback.

Next consider the *controlled-phase-query oracle*  $\Lambda_{x_a}((-1)^f) = \Lambda_{x_a}(e^{-i\pi H_f}) = (-1)^{f \wedge x_a}$  derived from (30). Observe that if we can query  $\Lambda_{x_a}((-1)^f)$  then we can implement  $G_f$  using a single-bit QPE, which requires two Hadamard gates applied to an ancilla qubit and a single  $\Lambda_{x_a}((-1)^f)$  query as

$$H_a \Lambda_{x_a}((-1)^f) H_a |x\rangle|0\rangle_a = |x\rangle|0 \oplus f(x)\rangle_a = G_f|x\rangle|0\rangle_a. \quad (39)$$

Similarly, two bit queries can simulate  $\Lambda_{x_a}((-1)^f)$  using an ancilla qubit  $|0\rangle_b$  to store  $f(x)$  as

$$(I \otimes G_f) R_{Z_a}(-\frac{\pi}{2}) R_{Z_b}(-\frac{\pi}{2}) R_{Z_a Z_b}(\frac{\pi}{2}) (I \otimes G_f) |x_a\rangle_a |x\rangle|0\rangle_b = c' (-1)^{f(x) \wedge x_a} |x_a\rangle_a |x\rangle|0\rangle_b, \quad (40)$$

where the constant  $c'$  is an unimportant global phase. (Here the three rotations on the left of (40) are derived from simulating the Hamiltonian representing the function  $x_a \wedge x_b$  for time  $\pi$  as to implement the operator  $(-1)^{x_a \wedge x_b}$ , and the second application of  $G_f$  uncomputes the ancilla qubit.) Hence, we easily see that the oracles  $G_f$  and  $\Lambda_{x_a}(e^{-i\pi H_f}) = (-1)^{f(x) \wedge x_a}$  are computationally equivalent, and both are at least as powerful as the phase oracle  $(-1)^f$ . The results of this article may be similarly applied to the study of complexity and reductions between further classes of oracles.

## 5 DISCUSSION AND FUTURE WORK

We have shown explicit rules and results for constructing Hamiltonians representing Boolean and pseudo-Boolean functions, including important classes of objective functions for combinatorial optimization. Applications include quantum gate-model and quantum annealing approaches to optimization, the simulation of diagonal Hamiltonians, and the construction of penalty or mixer Hamiltonians for problems with hard constraints. Moreover, we have shown our results entail quantum circuit constructions for controlled unitaries, in particular, those that reversibly compute a Boolean function in an ancilla register. The goal of these results is to give a toolkit which can be used generally towards the design and implementation of a wide range of quantum approaches for optimization, for related applications such as machine learning [55], and beyond. Our results give a unified view of existing problem mappings in the literature, such as those of [28, 40].

There are a variety of enticing applications and extensions of our results, and we briefly outline several additional directions. We emphasize that Fourier analysis is generally a very rich topic in computer science, mathematics, and physics, in addition to its many applications in quantum computing. A promising research direction is to further apply these tools, in particular more advanced ideas from the Fourier analysis of Boolean functions as applied in classical computer science, to the design and analysis of quantum algorithms. As mentioned, we leave a detailed analysis of general Hamiltonians acting on qubits or qudits as a topic of future work. A next step is to further classify *unfaithful* Hamiltonian representations of Boolean functions, where  $n$  variables are encoded in  $n' > n$  qubits, which in particular is an important paradigm for embedding problems on physical quantum annealing hardware, and, more generally, is related to the theory of quantum error correcting codes. Moreover, our results may have useful applications to quantum statistical mechanics, where many important Hamiltonians are given as linear combination of Pauli operators, such as the Ising or quantum XY models [5, 38]. Furthermore, exploring connections to quantum complexity theory may be fruitful, such as Hamiltonian complexity [19, 25, 36], or the computational power of restricted classes of quantum circuits [14, 15, 37]. Finally, it of interest whether techniques from quantum computing and quantum information can shed further insight on important open problems in classical computer science [23, 27].

## REFERENCES

- [1] Andris Ambainis. 2006. Polynomial degree vs. quantum query complexity. *Journal of Computer and System Sciences* 72, 2 (2006), 220–238.
- [2] Matthew Amy, Parsiad Azimzadeh, and Michele Mosca. 2018. On the controlled-NOT complexity of controlled-NOT-phase circuits. *Quantum Science and Technology* 4, 1 (2018), 015002.
- [3] Sanjeev Arora and Boaz Barak. 2009. *Computational Complexity - A Modern Approach*. Cambridge University Press. I–XXIV, 1–579 pages.
- [4] Giorgio Ausiello, Pierluigi Crescenzi, Giorgio Gambosi, Viggo Kann, Alberto Marchetti-Spaccamela, and Marco Protasi. 2012. *Complexity and Approximation: Combinatorial Optimization Problems and their Approximability Properties*. Springer Science & Business Media.
- [5] Francisco Barahona. 1982. On the computational complexity of Ising spin glass models. *Journal of Physics A: Mathematical and General* 15, 10 (1982), 3241.
- [6] Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A. Smolin, and Harald Weinfurter. 1995. Elementary gates for quantum computation. *Physical Review A* 52, 5 (1995), 3457.
- [7] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald De Wolf. 2001. Quantum lower bounds by polynomials. *Journal of the ACM* 48, 4 (2001), 778–797.
- [8] Richard Beigel. 1993. The polynomial method in circuit complexity. In *Proceedings of the 8th Structure in Complexity Theory Conference*. IEEE, 82–95.
- [9] J. D. Biamonte. 2008. Nonperturbative k-body to two-body commuting conversion Hamiltonians and embedding problem instances into Ising spins. *Physical Review A* 77, 5 (2008), 052331.

- [10] Zhengbing Bian, Fabian Chudak, William G. Macready, and Geordie Rose. 2010. The Ising Model: Teaching an old Problem New Tricks. *D-Wave Systems*, (2010).
- [11] Garrett Birkhoff and John Von Neumann. 1936. The logic of quantum mechanics. *Annals of Mathematics* 37, 4 (1936), 823–843.
- [12] Sergio Boixo, Vadim N. Smelyanskiy, and Hartmut Neven. 2017. Fourier analysis of sampling from noisy chaotic quantum circuits. arXiv:1708.01875 Retrieved from <https://arxiv.org/abs/1708.01875>.
- [13] Endre Boros and Peter L. Hammer. 2002. Pseudo-Boolean optimization. *Discrete Applied Mathematics* 123, 1 (2002), 155–225.
- [14] Sergey Bravyi, David Gosset, and Robert König. 2018. Quantum advantage with shallow circuits. *Science* 362, 6412 (2018), 308–311.
- [15] Michael J. Bremner, Ashley Montanaro, and Dan J. Shepherd. 2017. Achieving quantum supremacy with sparse and noisy commuting quantum computations. *Quantum* 1 (2017), 8. <https://quantum-journal.org/papers/q-2017-04-25-8/>.
- [16] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. 2018. The power of block-encoded matrix powers: Improved regression techniques via faster hamiltonian simulation. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP'19)*, Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi. 33:1–33:14. DOI: [10.4230/LIPIcs.ICALP.2019.33](https://doi.org/10.4230/LIPIcs.ICALP.2019.33)
- [17] Andrew M. Childs. 2004. *Quantum Information Processing in Continuous Time*. Ph.D. Dissertation. Massachusetts Institute of Technology.
- [18] Isaac J. Crosson, Dave Bacon, and Kenneth R. Brown. 2010. Making classical ground-state spin computing fault-tolerant. *Physical Review E* 82, 3 (2010), 031106.
- [19] Toby S. Cubitt, Ashley Montanaro, and Stephen Piddock. 2018. Universal quantum hamiltonians. In *Proceedings of the National Academy of Sciences* 115, 38 (2018), 9497–9502.
- [20] Ronald De Wolf. 2008. A brief introduction to fourier analysis on the boolean cube. *Theory of Computing, Graduate Surveys* 1, 1 (2008), 1–20.
- [21] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. 2014. A quantum approximate optimization algorithm. arXiv:1411.4028 Retrieved from <https://arxiv.org/abs/1411.4028>.
- [22] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. 2000. Quantum computation by adiabatic evolution. arXiv:quant-ph/0001106 Retrieved from <https://arxiv.org/abs/quant-ph/0001106>.
- [23] Yuval Filmus, Hamed Hatami, Steven Heilman, Elchanan Mossel, Ryan O'Donnell, Sushant Sachdeva, Andrew Wan, and Karl Wimmer. 2014. Real analysis in computer science: A collection of open problems. Preprint.
- [24] Michael R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY.
- [25] Sevag Gharibian, Yichen Huang, Zeph Landau, and Seung Woo Shin. 2015. Quantum hamiltonian complexity. *Foundations and Trends in Theoretical Computer Science* 10, 3 (2015), 159–282.
- [26] Steven Givant and Paul Halmos. 2008. *Introduction to Boolean Algebras*. Springer Science & Business Media.
- [27] Yingfei Gu and Xiao-Liang Qi. 2019. Majorana fermions and the sensitivity conjecture. arXiv:1908.06322 Retrieved from <https://arxiv.org/abs/1908.06322>.
- [28] Stuart Hadfield, Zhihui Wang, Bryan O’Gorman, Eleanor G. Rieffel, Davide Venturelli, and Rupak Biswas. 2019. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms* 12, 2 (2019), 34.
- [29] Stuart Hadfield, Zhihui Wang, Eleanor G. Rieffel, Bryan O’Gorman, Davide Venturelli, and Rupak Biswas. 2017. Quantum approximate optimization with hard and soft constraints. In *Proceedings of the 2nd International Workshop on Post Moores Era Supercomputing*. Association for Computing Machinery, New York, NY, 15–21. DOI: <https://doi.org/10.1145/3149526.3149530>
- [30] Peter L. Hammer and Sergiu Rudeanu. 2012. *Boolean Methods in Operations Research and Related Areas*. Vol. 7. Springer Science & Business Media.
- [31] Itay Hen and Marcelo S. Sarandy. 2016. Driver hamiltonians for constrained optimization in quantum annealing. *Physical Review A* 93, 6 (2016), 062312.
- [32] Itay Hen and Federico M. Spedalieri. 2016. Quantum annealing for constrained optimization. *Physical Review Applied* 5, 3 (2016), 034007.
- [33] Tad Hogg and Dmitriy Portnov. 2000. Quantum optimization. *Information Sciences* 128, 3–4 (2000), 181–197.
- [34] Tadashi Kadowaki and Hidetoshi Nishimori. 1998. Quantum annealing in the transverse Ising model. *Physical Review E* 58, 5 (1998), 5355.
- [35] Jeff Kahn, Gil Kalai, and Nathan Linial. 1988. The influence of variables on boolean functions. In *Proceedings of the 29th IEEE Symposium on Foundations of Computer Science*. IEEE, 68–80.
- [36] Julia Kempe, Alexei Kitaev, and Oded Regev. 2006. The complexity of the local hamiltonian problem. *SIAM Journal on Computing* 35, 5 (2006), 1070–1097.



- [37] Emanuel Knill and Raymond Laflamme. 1998. Power of one bit of quantum information. *Physical Review Letters* 81, 25 (1998), 5672.
- [38] Elliott Lieb, Theodore Schultz, and Daniel Mattis. 2004. Two soluble models of an antiferromagnetic chain. *Condensed Matter Physics and Exactly Soluble Models*. Springer, 543–601.
- [39] Nathan Linial, Yishay Mansour, and Noam Nisan. 1993. Constant depth circuits, Fourier transform, and learnability. *Journal of the ACM* 40, 3 (1993), 607–620.
- [40] Andrew Lucas. 2014. Ising formulations of many NP problems. *Frontiers in Physics* 2, 5 (2014), 1–15.
- [41] Florence Jessie MacWilliams and Neil James Alexander Sloane. 1977. *The Theory of Error Correcting Codes*. Vol. 16. Elsevier.
- [42] Catherine C. McGeoch. 2014. Adiabatic quantum computation and quantum annealing: Theory and practice. *Synthesis Lectures on Quantum Computing* 5, 2 (2014), 1–93.
- [43] Ashley Montanaro and Tobias J. Osborne. 2010. Quantum boolean functions. *Chicago Journal of Theoretical Computer Science* 2010, 1 (2010).
- [44] Michael Nielsen and Isaac Chuang. 2000. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge UK.
- [45] Michael A. Nielsen. 1997. Computable functions, quantum measurements, and quantum dynamics. *Physical Review Letters* 79, 15 (1997), 2915.
- [46] Noam Nisan and Mario Szegedy. 1994. On the degree of boolean functions as real polynomials. *Computational Complexity* 4, 4 (1994), 301–313.
- [47] Hidetoshi Nishimori. 2001. *Statistical Physics of Spin Glasses and Information Processing: An Introduction*. Vol. 111. Clarendon Press.
- [48] Ryan O'Donnell. 2014. *Analysis of Boolean Functions*. Cambridge University Press.
- [49] Masayuki Ohzeki. 2020. Breaking limitation of quantum annealer in solving optimization problems under constraints. *Scientific Reports* 10, 1 (2020), 1–12.
- [50] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy L. O'Brien. 2014. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications* 5, 1 (2014), 1–7.
- [51] Norbert Schuch and Jens Siewert. 2003. Programmable networks for quantum algorithms. *Physical Review Letters* 91, 2 (2003), 027902.
- [52] Jacob T. Seeley, Martin J. Richard, and Peter J. Love. 2012. The bravyi-kitaev transformation for quantum computation of electronic structure. *The Journal of Chemical Physics* 137, 22 (2012), 224109.
- [53] Mathias Soeken and Martin Roetteler. 2020. Quantum circuits for functionally controlled NOT gates. In *IEEE International Conference on Quantum Computing and Engineering (QCE'20)*. IEEE, 366–371.
- [54] Attila Szabo and Neil S. Ostlund. 1996. *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*. Dover.
- [55] Guillaume Verdon, Michael Broughton, and Jacob Biamonte. 2017. A quantum algorithm to train neural networks using low-depth circuits. arXiv:1712.05304 Retrieved from <https://arxiv.org/abs/1712.05304>.
- [56] Jonathan Welch, Daniel Greenbaum, Sarah Mostame, and Alán Aspuru-Guzik. 2014. Efficient quantum circuits for diagonal unitaries without ancillas. *New Journal of Physics* 16, 3 (2014), 033040.
- [57] James Daniel Whitfield, Mauro Faccin, and JD Biamonte. 2012. Ground-state spin logic. *Europhysics Letters* 99, 5 (2012), 57004.
- [58] Peter Woit. 2017. *Quantum Theory, Groups and Representations: An Introduction*. Springer.

Received August 2020; revised July 2021; accepted July 2021