The Eleventh International Symposium on Combinatorial Search
(SoCS 2018)

On the Complexity of Quantum Circuit Compilation

Adi Botea, Akihiro Kishimoto, Radu Marinescu
IBM Research, Ireland

Abstract

Quantum circuit compilation (QCC) is an important problem in the emerging field of quantum computing. The problem has a strong relevance to combinatorial search, as solving approaches recently published include constraint programming and temporal planning. In this paper, we focus on a complexity analysis of quantum circuit compilation. We formulate a makespan optimization problem based on QCC, and prove that the problem is NP-complete. To the best of our knowledge, this is the first study on the theoretical complexity of QCC.

1 Introduction

In quantum computing information is stored in qubits which are the basic memory units of quantum processors. Quantum algorithms typically process this information by applying quantum operations (called quantum gates) on subsets of qubits. Quantum gates are analogous to instructions on registers in classical computing. In general, a quantum algorithm must be compiled into a set of elementary quantum gates which are then applied in sequence at specific times in order to run on a specific quantum hardware (Nielsen and Chuang 2010; Rieffel and Pollack 2011).

Current computation models for quantum computing require the quantum algorithms to be specified as quantum circuits on idealized hardware because the physical hardware may have varying constraints. However, compiling a circuit for idealized hardware requires adding additional gates that move qubit states to locations where the desired gate could act upon them under the physical constraints of the actual quantum processor. Furthermore, quantum hardware suffers from a phenomenon called *decoherence* which degrades the performance of quantum algorithms over time. Therefore, it is also important to minimize the duration (i.e., the *makespan*) of the circuit that carries out the quantum computation in order to minimize the decoherence experienced by the computation (Nielsen and Chuang 2010).

There has been significant progress over the past few years on synthesizing quantum circuits from algorithm specification (Smith, Curtis, and Zeng 2016; Steiger, Haner, and Troyer 2016). However, the problem of compiling idealized quantum circuits for realistic quantum hardware has been explored only recently (Venturelli et al. 2017; Booth et al. 2018). This has lead to novel compilation methods based on classical planning and constraint programming techniques. More specifically, temporal planning is applied to the quantum circuit compilation problem to find solutions (plans) that minimize the duration of the corresponding compiled circuits. Subsequently, the planning solutions can be used by a constraint programming model to find a much higher quality solution, thus reducing further the quantum circuit's duration.

In this paper, we revisit the quantum circuit compilation (QCC) problem and explore some of its computational complexity aspects. Specifically, we consider the makespan optimization QCC problem on general graphs. We show that the problem is NP-complete. The NP-completeness of makespan optimal QCC with a planar graph remains open. Our NP-completeness proof holds even in the presence of additional features such as crosstalk constraints, which are present in certain existing quantum hardware architectures (Booth et al. 2018).

The paper is organized as follows. In Section 2 we formally define the QCC problem, and formulate the corresponding makespan optimization problem for general graphs. Section 3 presents our main result that proves the NP-completeness, while Section 4 provides concluding remarks and outlines a few directions of future work.

2 Problem Definition

We present a definition of Quantum Circuit Compilation (QCC) adapted from Booth et al.'s work (2018). We define the problem as a structure $\langle G = (V, E), A, I, g\rangle$. G is an undirected graph with several types of edges. (We say that the edges are colored. Edge types are explained later in this section.) Booth et al. (2018) state that graphs are planar. In our work we do not address this condition.

The time is discretized.

Nodes in the graph represent qubits. A is a set of qubit states with $|A| \leq |V|$. At any point in time, there is a configuration where each qubit state occupies a node in the graph (qubit). A node can host at most one qubit state at a time. We say that a qubit without a qubit state $q \in A$ has an empty qubit state. The initial configuration (i.e., the qubit state of each qubit at the beginning) is represented by I.

Edges correspond to 2-qubit quantum gates. We consider *swap* gates and *phase separation* (PS) gates (Venturelli et al. 2017; Farhi, Goldstone, and Gutmann 2014).

At a given time, the qubit states of two adjacent qubits can be involved in a gate operation. If two qubits n_1 and n_2 are connected through a swap edge (i.e., a swap gate is applied to qubits n_1 and n_2, respectively), the two qubit states can swap their locations. Such an operation can take several discrete steps, given as an input in the definition of the problem. Swap operations include the particular case when one of the qubits (e.g., n_2) has an empty qubit state (which can happen when $|A| < |V|$). In such a case, n_1's qubit state moves to n_2, and the qubit state of n_1 becomes empty.

Likewise, two qubits connected via a PS edge can be involved in a PS gate operation. PS operations are used to define the goal, as shown later in this section. For each PS gate, the duration of such an operation is provided as an input as well.

The type of a gate (i.e., swap or PS) together with its corresponding duration uniquely define the type (the color) of the corresponding edge in our graph G.

Operations can be performed in parallel, provided that they do not interfere with each other. That is, at any given time, a qubit may be involved in at most one operation.

The goal g is defined as a set of pairs of qubit states. If a pair of qubit states (q_i, q_j) belongs to g, then at some point in time q_i and q_j must perform a PS operation together. As mentioned earlier, to be able to perform a PS operation, we need to bring the two qubit states at hand to two nodes (qubits) connected through a PS edge (gate).

A solution is a series of operations that leads to satisfying all goals (not necessarily all at the same time). A solution is optimal if the makespan (i.e., total number of time steps with moves allowed in parallel) is minimal.

As an extension to the problem, Booth et al. (2018) present the so-called *crosstalk constraints* (QCC-X), which are present in certain quantum hardware architectures, such as Google devices (Boxio 2016). According to these constraints, when a qubit is involved in a gate operation, its adjacent qubits cannot be involved in a different operation at the same time. Consider two qubits n_1 and n_2 performing a swap or a PS operation. If n_3 is adjacent to n_2, then n_3 cannot be involved in a gate operation (swap or PS operation) at the same time. Our theoretical results, presented in the next section, hold with and without the QCC-X constraints.

QCC is related to the problem of swapping colored tokens in graphs (Yamanaka et al. 2015). In this problem, the nodes of a graph can host one colored token each. Two adjacent tokens can swap their positions. Each node has a goal color. The task is to reach a configuration where the goal color of each node coincides with the color of its hosted token. Significant differences between QCC and the token swapping problem include the way the goal is defined, and the fact that tokens with the same color are indistinguishable in token swapping problems. See (Yamanaka et al. 2015) for complexity results available for various formulations of token swapping problems.

QCC also bears a relation to multi-agent path planning (Kornhauser, Miller, and Spirakis 1984; Silver 2005). If we view qubit states as agents, the problem involves moving agents in a graph. There are important differences, however. In multi-agent path planning, swapping the positions of two agents along an edge is not permitted. Furthermore, the goal is to bring each agent to its target node. The target of each agent is fixed. In contrast, in QCC, we can perform a goal PS operation at any PS gate (edge) available in the graph.

3 Problem Complexity

Definition 1 (MO-QCC)**.** *The MO-QCC (Makespan Optimal QCC) is defined as follows. Input: A QCC instance as defined in the previous section, with no crosstalk constraints; and a positive integer τ. Question: Does the instance have a solution with the makespan no larger than τ?*

Definition 2 (MO-QCCX)**.** *The MO-QCCX (Makespan Optimal QCC with crosstalk constraints) is similar to MO-QCC, except that crosstalk constraints must be satisfied.*

We prove that both MO-QCC and MO-QCCX are NP-complete. For clarity, we focus first on MO-QCC, and refer to MO-QCCX in the last part of this section. The proof idea is related to proofs available for multi-agent path planning problems (Surynek 2010; Yu and LaValle 2015). The hardness will be shown with a reduction from SAT. The reduction will make use of two types of gadgets, called *variable gadgets* and *clause gadgets*, respectively. Next, we introduce each gadget type.

3.1 Variable Gadgets

A variable gadget is a QCC instance. Such a gadget can be defined for any number n of qubit states. We introduce variable gadgets with an example for $n = 4$, illustrated in Figure 1. Then we generalize to arbitrary n values. In the example, we have 4 qubit states, $A = \{q_1, q_2, q_3, q_4\}$. The graph topology is shown in the figure. The edge types are explained in the caption. Initially, the qubit states are at the nodes in the middle "column" of the graph topology. The goal is $\{(q_1, q_2), (q_2, q_3), (q_3, q_4)\}$. The generalization from $|A| = 4$ to an arbitrarily large set of agents $A = \{q_1, \ldots, q_n\}$ is easy: have n "rows" in the graph shown in Figure 1. Each "row" is one edge longer than the previous in each direction, except for the last row, which has the same length as the second last one. Set the goal to $\{(q_1, q_2), (q_2, q_3) \ldots (q_{n-1}, q_n)\}$.

Given a qubit state q and a time t, we say that q is idle at time t if q is not involved in any gate operation (swap or PS) at that time.

We say that a solution has no *early idle qubit states* if no qubit state stays idle at any time before fulfilling its goal PS operations.

Lemma 1. *Consider a variable gadget with $|A| = n$. There exists a solution with no early idle qubit states, whose makespan is $d(2n + z - 1)$.*

Proof. This can be shown by induction on n. The claim is straightforward for $n = 2$: the two qubit states require $d(2 + z)$ time steps to reach the PS gate, and another d time steps to complete the PS operation, to a total of $d(z+3) = d(2n+$

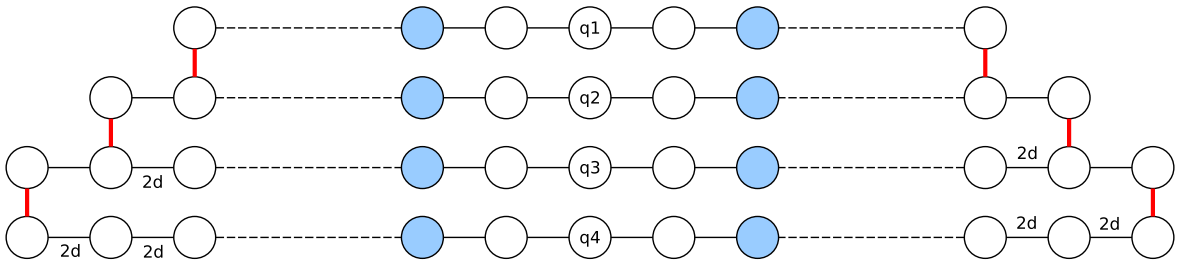

Figure 1: An example of a variable gadget, used as a running example for Lemma 1. A thin solid line represents a swap edge in the graph. A dashed line represents a chain of z swap edges and $z-1$ nodes, where z is a parameter specific to each variable gadget. A thick vertical line (red on a colour printout) represents a PS edge. Swap edges with no label have a duration d. (This includes the swap edges contained in the dash-line chains.) A given "row" i has 0 or more swap edges located between the end of the dash-line chain and the first PS gate adjacent to the row. These have a duration of $2d$. Their duration is explicitly marked in the figure. In the initial configuration I, the qubit states q_1, q_2, q_3, q_4 are located at the nodes in the "middle column", as shown in the figure.

$z-1)$ time steps. See the top two rows in Figure 1 for an illustration.

For $n > 2$, assume that the result holds for $n-1$. Thus, in the subproblem corresponding to the first $n-1$ qubit states, q_{n-1} finishes its first PS operation by the time $d(2(n-1)+z-1)$. This qubit state needs d additional time steps to reach its second PS gate, increasing the time to $d(2(n-1)+z-1)+d = d(2n+z-2)$.

Observe that q_n can reach the PS gate by the same time, with no idle times: $d(2+z)+2d(n-2) = d(2+z+2n-4) = d(2n+z-2)$. The term $d(2+z)$ corresponds to the part where q_n travels to the end of the dash-line chain. The term $2d(n-2)$ is for the remaining swap operations.

The PS operation for q_{n-1} and q_n requires d time steps, increasing the makespan to $d(2n-2+z)+d$.

In summary, the solution to the $n-1$ subproblem together with the actions involving q_n form a solution with no early idle qubit states. The makespan is $d(2n+z-1)$. □

Lemma 2. *Consider a variable gadget with $|A| = n$, and with each dash-line chain having z edges. Then an optimal solution has $d(2n+z-1)$ time steps.*

Proof. A solution with the corresponding makespan value exists, according to Lemma 1. It remains to show that every solution requires at least $d(2n+z-1)$ time steps.

Observe that the qubit state q_n needs at least $d(2+z)+2d(n-2)+d = d(2n+z-1)$ time steps to fulfill its goal PS operation, as shown in the proof to Lemma 1. Finally, the PS operation for q_{n-1} and q_n requires d time steps. Therefore, the makespan of any solution is no smaller than $d(2n+z-1)$.

□

Lemma 3. *In a variable gadget, no makespan optimal solution can possibly have early idle qubit states.*

Proof. Assume by contradiction that a qubit state q_i can stay idle before fulfilling its goal PS operations. If $i = n$, it immediately follows that the makespan gets larger than the value presented in Lemma 2, which is a contradiction.

Assume now that $i < n$. As q_i and q_{i+1} synchronize for their common PS operation, it follows that q_{i+1} has an early idle time. It recursively follows that q_n has an early idle time, which takes us to the case $i = n$, addressed earlier. □

Given a variable gadget with n qubit states, define S_l as the set of blue nodes[1] at the left (two columns away from the middle column), and S_r as the set of blue nodes at the right. Clearly, $|S_l| = |S_r| = |A| = n$.

Corollary 1. *In any makespan optimal solution of a variable gadget, at time $t = 2d$ either all qubit states $q \in A$ are located in S_l, or they are all located in S_r. Both options are possible, in the sense that there exists an optimal solution involving subset S_l, and there exists an optimal solution involving S_r.*

Proof. In a makespan optimal solution, we have that either: i) all qubit states travel to the left and use the PS gates at the left; or ii) all qubit states travel to the right and use the PS gates at the right. According to Lemma 3, qubit states cannot have early idle times in an optimal solution. It follows that, at time $2d$, either all qubit states are in S_l (case i) or all qubit states are in S_r (case ii). □

3.2 Clause Gadgets

A clause gadget is a QCC instance as illustrated in Figure 2. There are two qubit states, a_1 and b_1, whose initial locations are shown in the picture. The node initially hosting a_1 branches into m neighbor nodes. We say that the *degree* of the clause gadget is m. The goal is (a_1, b_1). Swap and PS operations require d time steps each.

Lemma 4. *In a makespan optimal solution of a clause gadget, the two qubit states have no early idle times. At time $2d$, the qubit state a_1 is at one of the m blue nodes (light grey on a black and white printout).*

[1] Light grey nodes on a black and white printout.

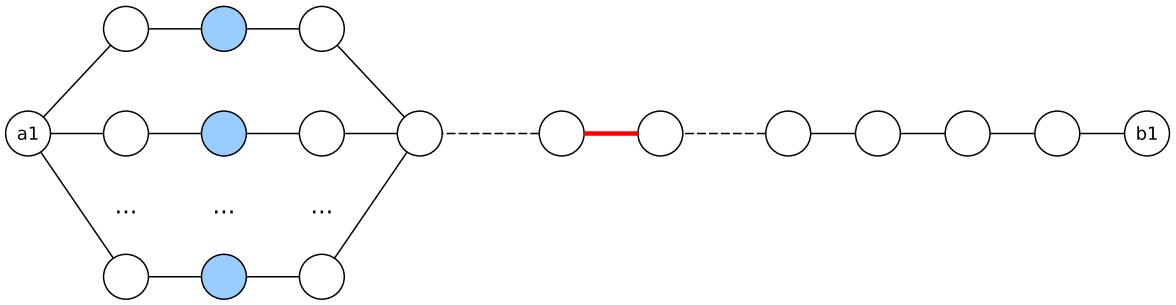

Figure 2: A clause gadget. As in Figure 1, thin solid lines are swap edges. A dashed line is a chain of z swap edges and $z-1$ nodes, where z is a parameter specific to each gadget. The thick red line at the middle is a PS edge.

Proof. The number of swap operations needed to reach the PS gate is the same for both qubit states, which is why early idle times cannot be present in an optimal solution. □

Lemma 5. *The makespan of an optimal solution is $d(5+z)$, where z is the number of edges in a dash-line chain.*

Proof. The two qubit states need $d(4+z)$ time steps to travel in parallel to the PS gate. The PS operation requires d additional time steps. □

3.3 Hardness Results

Theorem 1. *MO-QCC is NP-complete.*

Proof. First, we argue that the problem is in NP. Observe that QCC instances can be solved suboptimally in polynomial time as follows. Pick a pair of qubit states in the goal, have the two qubit states meet at a PS gate, and apply a PS operation. Repeat this for all pairs of qubit states in the goal. It follows that optimal solutions have a polynomial makespan. Thus, any optimal solution can be verified in polynomial time.

The hardness is shown with a reduction from SAT. Consider an arbitrary SAT formula in the conjunctive normal form (CNF). Extend it into an equivalent formula where, for each variable v, the number of positive literals of v is equal to the number of negative literals of v. This can be performed as follows: assume that initially v has o_p occurrences as a positive literal and o_n occurrences as a negative literal, with $o_p \neq o_n$. If $o_n < o_p$, add a clause $(v \vee \neg v \vee \cdots \vee \neg v)$, with $o_p - o_n + 1$ copies of $\neg v$ in that clause. If $o_p < o_n$, add a clause $(\neg v \vee v \vee \cdots \vee v)$, with $o_n - o_p + 1$ copies of v in that clause.

Given a variable v, let n_v be the number of its positive literals, which is equal to the number of its negative literals. Consider an arbitrary but fixed ordering of v's positive literals, OP_v, and an arbitrary but fixed ordering of the negative literals, ON_v.

Construct a QCC instance α as follows.

We build a variable gadget for each variable in the SAT formula, and a clause gadget for each clause.

For variable v, build a variable gadget $VG(v)$, where the set A contains n_v qubit states, denoted as $q_1^v, \ldots, q_{n_v}^v$. Nodes in S_l correspond to negative literals of v. We label these nodes as $N_1^v, \ldots, N_{n_v}^v$ (in the order given by OP_v). Nodes in S_r correspond to the positive literals of v. We label these nodes as $P_1^v, \ldots, P_{n_v}^v$ (in the order given by ON_v).

For each clause c with m literals, build a clause gadget $CG(c)$ with degree m. Call the two qubit states a_c and b_c.

Recall that in an individual gadget all dash-line chains have the same length. Set the length of the chains (i.e., the z value in each gadget) in such a way that all gadgets (both variable and clause gadgets), taken as independent instances, have the same optimal solution makespan.[2] Let τ be such a common optimal makespan.

So far, we have built a set of independent gadgets. Now we combine them to obtain our instance α. The graph of the instance α is the union of the graphs of all gadgets, where the gadgets may have some overlapping nodes. Specifically, a variable gadget and a clause gadget may have some common blue nodes. If the k-th negative literal of variable v belongs to a clause c, the corresponding node N_k^v belongs both to $VG(v)$ and $CG(c)$. If the l-th positive literal belongs to a clause c', the node P_l^v belongs to both $VG(v)$ and $CG(c')$.

The set of qubit states is the union of all qubit states from all gadgets, the goal is the union of all goals, and the initial configuration is the union of all initial configurations. This completes the construction of α.

Having a qubit state q_i^v in a blue node at time $2d$ is equivalent to setting the literal corresponding to that node to false. Having a qubit state a_c in a blue node (corresponding to a literal in the clause c) at time $2d$ ensures that the corresponding literal can be set to true in the SAT formula.

We claim that the SAT formula has a solution if and only if our QCC instance has a solution of makespan τ. If the SAT formula has a solution, guide all q_i^v qubit states corresponding to a variable v through P^v nodes if v = false in the solution, or through N^v nodes otherwise. As each

[2] To achieve this, use the formulas provided in Lemmas 2 and 5. That is, for a SAT formula with R variables $v_1, \ldots, v_R$, and M clauses $c_1, \ldots, c_M$, we impose $2n_{v_1} + z_{v_1} - 1 = 2n_{v_2} + z_{v_2} - 1 = \cdots = 2n_{v_R} + z_{v_R} - 1 = 5 + z_{c_1} = \cdots = 5 + z_{c_M}$. Define $\mu = \arg\max_{i=1}^{R} n_{v_i}$. Set $z_\mu = 5$, and set all other z values so that all equalities hold. Observe that the resulting graph remains polynomial in the size of the input.

clause c has at least one true literal, the qubit state a_c can use the corresponding blue node to progress with no interference with other qubit states (from an overlapping variable gadget). Thus, each gadget is solved in τ time steps, which further implies that the instance α is solved in τ time steps.

If the QCC instance has a solution of τ time steps (that is, an optimal solution), it follows all qubit states a_c and q_i^v are located at blue nodes at time $2d$, according to Corollary 1 and Lemma 4. Set to true literals corresponding to blue nodes where qubit states a_c are located at time $2d$. This ensures that all clauses in the SAT formula hold. In any solution of τ steps, at time $2d$, qubit states $q_1^v, \ldots, q_{n_v}^v$ are either all in $N_1^v, \ldots, N_{n_v}^v$, or all in $P_1^v, \ldots, P_{n_v}^v$, according to Corollary 1. This ensures the so-called Boolean consistency of the variable v (i.e., either all positive literals are true and all negative literals are false; or all positive literals are false and all negative literals are true). The Boolean consistency together with the fact that each clause is true result in a valid solution to the SAT instance. □

Corollary 2. *MO-QCCX is NP-complete.*

This result holds because the proof presented in this section never infringes any crosstalk constraints.

Corollary 3. *MO-QCC and MO-QCCX are NP-complete even when all swap gates have the same duration.*

We can easily adapt the proof for this extra condition, by replacing a swap gate with the duration $2d$ with a chain of two gates with the duration d each.

4 Conclusion and Future Work

Quantum circuit compilation is an important problem, with a strong relevance to both quantum computing and combinatorial search. Despite this, no complexity analysis of the problem has been available. In this paper, we have provided a first study of the problem hardness. We have formulated a makespan optimization problem, based on QCC on general graphs, and have proven that the problem is NP-complete. The result holds even in the presence of crosstalk constraints.

In future work we plan to investigate the problem hardness under the additional condition that the graph is planar, a common feature in current quantum hardware architectures. In addition, we plan to study the existence of approximation algorithms that return solutions with bounded suboptimality guarantees.

Acknowledgment

Jeremy Frank has introduced us to the QCC problem in the Dagstuhl Seminar 18071 "Planning and Operations Research". We thank Jeremy Frank and Minh Do for discussions on this topic, and the anonymous reviewers for their feedback. We thank Inge Vejsbjerg for proofreading our paper.

References

Booth, K. E. C.; Do, M.; Beck, C.; Rieffel, E.; Venturelli, D.; and Frank, J. 2018. Comparing and integrating constraint programming and temporal planning for quantum circuit compilation. In *Proceedings of the International Conference on Automated Planning and Scheduling*.

Boxio, S. 2016. Characterizing quantum supremacy in near-term devices. In *arXiv preprint arXiv:1608.0026*.

Farhi, E.; Goldstone, J.; and Gutmann, S. 2014. A quantum approximate optimization algorithm. In *arXiv preprint arXiv:1411.4028*.

Kornhauser, D.; Miller, G.; and Spirakis, P. 1984. Coordinating pebble motion on graphs, the diameter of permutation groups, and applications. In *Proceedings of the 25th Annual Symposium on Foundations of Computer Science (FOCS)*, 241–250. IEEE Computer Society.

Nielsen, M., and Chuang, I. 2010. *Quantum computation and quantum information*. Cambridge University Press.

Rieffel, E., and Pollack, W. 2011. *Quantum computing: a gentle introduction*. MIT Press.

Silver, D. 2005. Cooperative pathfinding. In *Proceedings of the 1st Artificial Intelligence and Interactive Digital Entertainment Conference, AIIDE 2005*, 117–122.

Smith, R.; Curtis, M.; and Zeng, W. 2016. A practical quantum instruction set architecture. In *arXiv preprint arXiv:1608.03355*.

Steiger, D.; Haner, T.; and Troyer, M. 2016. Projectq: an open source software framework for quantum computing. In *arXiv preprint arXiv:1612.08091*.

Surynek, P. 2010. An optimization variant of multi-robot path planning is intractable. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA*.

Venturelli, D.; Do, M.; Rieffel, E.; and Frank, J. 2017. Temporal planning for compilation of quantum approximate optimization circuits. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 4440–4446.

Yamanaka, K.; D. Demaine, E.; Ito, T.; Kawahara, J.; Kiyomi, M.; Okamoto, Y.; Saitoh, T.; Suzuki, A.; Uchizawa, K.; and Uno, T. 2015. Swapping labeled tokens on graphs. *Theoretical Computer Science* 586:81–94.

Yu, J., and LaValle, S. M. 2015. Optimal multi-robot path planning on graphs: Structure and computational complexity. *CoRR* abs/1507.03289.