Abstract Primitives for Quantum Computation

Luca Mondada

Department of Computer Science, University of Oxford
luca.mondada@cs.ox.ac.uk

10th October 2021

Abstract

The quantum circuit language has become the de facto *standard to specify quantum computations both for algorithm design and for execution on hardware. However, as users start exploring more complex computational problems, more abstract languages and primitives will become essential. Simultaneously, hardware providers and compilers are keen to take advantage of architecture- and problem-specific optimisation opportunities, which are hard to express in the circuit language.*

New quantum primitives are necessary that provide both a useful abstraction to the user and optimisation opportunities to the quantum compiler. We contribute to this effort in two ways. In the first part, we present a systematic survey of previous work, providing an overview of the abstract quantum primitives that have been proposed. From our conclusions, we develop in the second part a new additive model of quantum computation that aims to address some of the issues of existing primitives. We show how this new model provides new optimisation opportunities and yields new insight into quantum algorithm design.

Contents

Introduction

The quantum circuit is an incredibly successful abstraction that has been widely adopted in quantum computing [1]. It can represent arbitrary computations and can be readily compiled for execution on current hardware. However, as the number of available qubits increases on devices and in applications, constructing circuits out of higher-level building blocks becomes a necessity.

Without higher-level gate primitives and abstractions, it will become steadily harder to manage circuits with hundreds to thousands of qubits. As the underlying Hilbert space grows exponentially, the current popular variational approaches will also quickly start to face scaling issues [2]. We are in a sense entering a long NISQ tunnel, in which quantum circuits as we know them will become unwieldy and our current classical compute-heavy design approaches will start to fail, but where

we have not reached the fault tolerant scale yet to make famous quantum algorithm proposals such as Shor's factoring [3] or Grover's search algorithm [4] viable.

Besides circuit design and convenience, there are also circuit optimisation-driven demands for new computing primitives. Limitations in current circuit optimisation performance as well as emerging hardware primitives and architectures are making it increasingly hard to make full use of hardware capabilities. Multi-qubit operations as they exist on certain ion trap devices [5, 6] are particularly hard to synthesise out of lower-level gate primitives.

In this paper, we tackle the problem of finding new abstractions for the design and optimisation of quantum circuits. To that end, this paper is split in two parts. In the first part, we present an extensive review of existing quantum structures and abstractions. We aim to be exhaustive by covering the litterature systematically by reviewing the over 200 combined results obtained from searches on multiple scientific databases.

From our obersvations and conclusions of the survey, we then proceed in the second part to proposing a new formulation of quantum computation based on additive Hilbert spaces, which we claim simplifies the mental model of quantum computation. Our *additive model* of quantum compuation is defined as a symmetric monoidal category and its diagrammatic description is developed.

Diagrammatic representations of quantum processes have a rich history [7, 8] and have heavily contributed to developping new understandings of quantum computation and new quantum computation optimisation techniques [9, 10]. We hope to present a new step in this direction. We describe how our proposed model could allow new compilation and quantum optimisation approaches, and how it could simplify quantum computation design.

We also discuss the limitations of the proposed additive model and explore avenues to mitigate these issues.

Part I
A Review of higher-order Quantum Primitives

1. Scope of the review

We aim to review past work that contributed to building what we call *higher-order NISQ primitives*: within the circuit model of quantum computation, these constructs should allow the user to think at a higher level of abstraction, whilst giving the compiler more freedom for optimisation given NISQ constraints. We define such primitives as any abstraction that can be useful to define quantum computations, with a few constraints, which we now detail.

Optimisation-relevant. The considered primitives must leave scope for optimisation: indeed, abstract primitives typically come with increased quantum cost. To remain useful in real life, we restrict our attention to proposals that have the potential for optimisations that could be leveraged by a quantum compiler.

Hardware compatible. Any primitive must be executable on realistic hardware, in so far as it should be possible to compile such a primitive at least in principle to the quantum circuit model for execution on hardware.

Higher-order abstraction. Finally, we want primitives that make thinking about quantum computing easier – we exclude for instance any gates that are introduces purely from hardware motivations.

A word about optimisation: while optimisation techniques are a key component of this review, this is not a review on quantum optimisation. We are only interested in optimisations in so far as they might enable abstract primitives to be used in computations on quantum hardware. We will discuss optimisation techniques when they are key to a particular

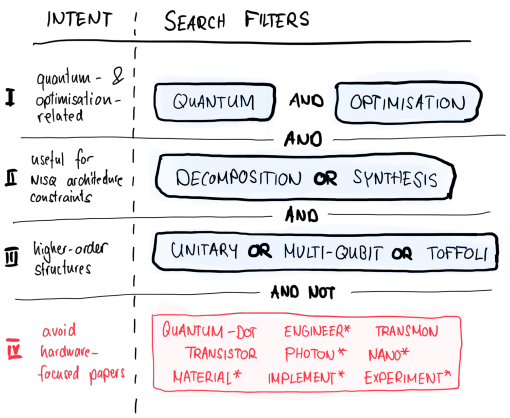


Figure 1: *Filters for the quantitative review paper search.*

primitive, but will not be exhaustive in this respect.

2. Paper selection

Our aim was to be as exhaustive and quantitative as possible. Our review includes all papers returned by searches on the major peer-reviewed paper databases as well as the arXiv. There were well over 200 papers in total.

i. Selection process

We performed exhaustive searches on Scopus (`scopus.com`), Web of Science (`webofscience.com`), IEEE Xplore (`ieeexplore.ieee.org`) and the arXiv (`arxiv.org`). We used the "advanced search" feature available on most platform to define an exact search query, designed to find all papers that could be relevant to our review. We have kept the scope of the search intentionally broad so as to not miss any significant paper. The query results are then filtered manually in a later step to keep only relevant results.

The search is structured as a conjuction of four search groups that we number I - IV. Figure 1 illustrates how the search is broken down into groups, where each group is assigned to a specific search intent. Groups I - III are whitelists that define the papers we are interested in, and reflect the three criterion described earlier, whereas group IV is a negative blacklist, introduced to filter out the large number of hardware-focused papers present in the raw results.

The whitelist filters accept search matches on all available metadata on the platforms Web of Science, IEEEXplore and arXiv. In Scopus searches, we made use of the search feature restricting queries to titles, abstract and keywords, which should contain the most relevant metadata.

To avoid discarding articles that mention hardware implementations – for instance in their abstract – whilst having a broader software scope, the blacklist only applies to paper titles. Finally, in the case of the arXiv, we only considered publications from the years 2020 and 2021, in the assumption that older relevant research would also be published in peer-reviewed venues.

ii. Search results

We obtain a total of 265 hits when merging the results of all platforms (Scopus: 103, WoS: 86, IEEE: 49, arXiv: 27). We import all entries into Jabref [11], automatically finding and removing 57 duplicates. We filter out the remaining papers by categorising the results manually into one of 7 topics from their titles and abstracts. We distinguish between papers deemed "on-topic" (103 results) and results in other quantum fields. Table 1 details the resulting statistics. The majority of papers filtered out in this stage were either duplicates (52 results) or hardware-related (18 results). This is expected as hardware papers historically form a large portion of publications in the domain. This fact gives us a certain confidence that our search successfully targetted the desired scope.

We thus obtain a set of 103 papers giving us an overview of proposed higher-order NISQ primitives. The results are presented in detail in the next section.

Table 1: *Filtering papers in search results.*

Category	Number of Papers
On topic	**103**
Hardware	18
Variational	9
Simulation	5
Chemistry	3
Routing	2
Duplicates	52
Other	16

3. Survey results

Table 2 provides a summary of the primitives encountered in the 103 selected papers. Two primitives stand out as being particularly dominant in the set of papers: reversible circuits and the matrix representation. We divide these papers into the different optimisation techniques used with these primitives. These give an idea of the potential optimisation techniques that these primitives enable. We reiterate however that the focus of this survey was not to make a exhaustive list of either reversible circuits optimisation or matrix synthesis. For the case of reversible circuits, a review was conducted by Saeedi et al in [106]. Although the review is aging quickly by now, it still gives a good overview of the different optimisation strategies in the field.

It is also interesting to note that many of the reversible circuit-related papers are older, with peak activity around 2013–2014. In contrast, the work on matrix representation has accelerated in the last years and has become an extremely active area of research. This can be understood as matrix decomposition techniques are especially promising in the low qubit counts of the current NISQ area; with the current stringent constraints in quantum hardware, it makes sense to trade additional cheap classical computational power for increased quantum capabilities. Remarkable progress has been made in the field, pushing the limit of feasible near-optimal circuit decompositions up to 6 qubits [68]. As qubit counts increase, however, matrix decompositions become increasingly unaffordable to perform classically.

We will discuss each of these two dominant representations in their own subsection below. A further 7 papers introduce novel primitives which we will discuss in more detail in a dedicated third subsection.

i. Reversible classical circuits

Reversible circuits correspond to the subset of quantum computations that can be obtained using classical logic [32]. In practice, this corresponds to computations given by matrices with entries that are only zeros and ones. With orthonormality, one can see that these matrices correspond exactly to the symmetric group S_{2^n} of all permutations of the basis vectors of the 2^n-dimensional Hilbert state space.

Our focus is on the different representations that reversible circuit primitives can be expressed in. These typically allow particular optimisations to be applied. For a detailed review of the optimisation techniques themselves, however, we refer to the papers referenced in table 2, as well as the above-mentioned survey paper on the topic.

Permutations can be viewed equivalently as the set of circuits generated by the CNT gate set, made of the NOT gate (uncontrolled), the CNOT gate (1-controlled) and the Toffoli (2-controlled) gate. Note that both in the classical and quantum case, any higher controlled n-Toffoli can be decomposed to the CNT gate set with standard decompositons [107], so that we can typically consider n-Toffolis to be part of the CNT gate set. In the special case of quantum, the standard, 2-controlled, Toffoli can itself be decomposed in terms of CNOTs and single unitary (non-classical) operations [108].

We can thus naturally split the reversible circuit synthesis problem in roughly two steps. Permutations are first decomposed into circuits in the generalised toffoli gate set, where arbitrarily controlled n-Toffolis are allowed. These circuits are then further decomposed into a universal set of quantum primitives, such as single-qubit rotations and CNOTs. This step

Table 2: *Summary of reviewed papers.*

Category	Number of Papers	References
Reversible circuits		
Templates and local optimisations	20	[12–31]
Search-based	13	[32–44]
ESOP-based	9	[45–53]
Swap-based	5	[54–58]
Decision Diagrams	3	[59–61]
Other approaches	4	[62–65]
Matrix representation		
Search-based	9	[66–74]
Linear Algebra decompositions	8	[75–82]
Template and local optimisation	3	[83–85]
Other various primitives		
Specialised Gate Decompositions	4	[86–90]
Ancilla management	3	[91–93]
Quantum Decision Diagram	1	[94]
Pauli gadgets	1	[95]
Quantum Karnaugh Map	1	[96]
Off topic	9	[97–105]

can be performed easily using the standard known decompositions for the n-Toffolis.

The first step is a purely classical optimisation: both reversible circuits and the target gate set can be expressed in classical logic. As a result of this, circuit synthesis in the n-Toffoli gate set is among the most well-studied and understood problems in quantum computing. The second step, on the other hand is purely quantum, as Toffolis can only be decomposed into two-qubit operations in the presence of the "square-root of NOT" gate, the quantum V-gate. We first look at the first problem: the classical problem of expressing reversible logic in the CNT gate set.

The number of reversible circuits grows exponentially and as such their description typically takes exponential space. The most straight-forward such representation is as a permutation matrix or a truth table. Such representations are ideal for swap-based, also known in the litterature as transformation-based, circuit synthesis. Typically, the circuit implementing the target permutation is gradually refined by appending gates, typically chosen by a greedy distance metric between the target permutation and the circuit. We prefer the term swap-based as it highlights how the reversible gates typically used (NOT, CNOT and n-Toffoli) act as permutation swapping pairs of outputs – a n-Toffoli gate on a n-qubit circuit will swap exactly two outputs, whereas an uncontrolled NOT gate on a single qubit will swap all outputs in pairs.

In one of the more recent approach, Susam and colleagues pre-compute and table the optimal circuit representations for every permutation that swaps two inputs and leave all other fixed [57]. The number of such permutations scales much more favourably compared to earlier attempts at finding and tabling optimal arbitrary permutations. These can then be used as part of a standard selection sort to synthesise arbitrary permutations. This allows for fast circuit synthesis up to 20+ qubits in a fraction of a second, with good performance.

Explicit truth table representations are also ideal for search-based optimisation strategies that perform searches in the circuit space to find optimal or near-optimal circuits implementing the desired reversible circuit. Using hash tables to check for collisions quickly, all

optimal 4-qubit permutations were generated in [36].

There are also other representations that although exponential in the worst case, provide in many cases very succinct descriptions of reversible circuits. A popular approach has been to express outputs of reversible circuits as binary arithmetic functions of the inputs, where products aare given by ANDs and nd additions by XORs. Such expressions can in particular be rewritten as exclusive sums of product terms (ESOP).

ESOP expressions can easily be translated into a circuit: product terms can be obtained with an n-Toffoli, and terms can be XORed with CNots. ESOPs are not unique but can often be expressed in canonical forms such as positive polarity Reed-Müller codes (PPRM): any ESOP over variables $x_1, \ldots, x_n$ can be expressed uniquely in its PPRM form as $a_1 x_1 \oplus a_2 x_1 x_2 \oplus \cdots \oplus a_n x_1 \cdots x_n$, with $a_1, \ldots, a_n \in \{0, 1\}$. ESOP-based circuit synthesis strategies then consist in finding an efficient order in which to compute product terms, so that computations and ancilla bits can be reused.

A third popular representation are Decision Diagrams. Decision Diagrams are computational graphs that can be obtained by decomposing ESOPs recursively into a tree structure. Each node combines the expressions of its children into a larger expression, with leaf nodes being variables or literals. Whereas ESOP-based circuit synthesis usually require a fixed number of ancilla qubits and a considerable circuit depth, the performance of decision diagram-based approaches is entirely commanded by the number of nodes in the tree. In practice, such approaches tend to use more ancillas but produce much shorter circuits. In recent work, Stojković and her colleagues have proposed a smart synthesis scheme based on decision diagrams that significantly reduced the number of ancillas required compared to previous work [61]. This makes decision diagrams one of the most promising avenue of work when there is a circuit-depth versus ancilla use trade-off to be made.

The second half – the non-classical half – of the reversible circuit synthesis problem, in which circuits in the CNT gate set must be further decomposed into a quantum gate set, is less well-studied. Most of the work has focused on expressing simplification rules for chains of Toffoli gates [21–28]. Mohammadi and Eshghi introduced 4-valued truth tables to perform circuit searches over a non-classical gate set, using controlled-V gates [44]. Soeken et al. as well as Rahman et al. incorporated controlled-V gates into template matching strategies and showed significant improvements in synthesised gate count [29, 30]. Another interesting approach is to decompose Toffolis only up to relative phase, introducing a lot of freedom in the quantum decompositions that are required [109], compared to the traditional classical decompositions.

Another feature of reversible circuit synthesis that is particularly relevant in the quantum case is the ancilla vs circuit depth trade-off that we touched on in Decision Diagrams. Different Toffoli decompositions with more or less ancillas are known to make use of this trade-off [110]. T-depth optimisation for reversible circuits using ancillas has also been presented in [111]. Finally, uncompute optimisation, very relevant to Toffoli decompositions and reversible circuits in general has been proposed in [93].

ii. Matrix decomposition

Unsurprisingly, the other popular primitive is the unitary matrix representation, which describes the quantum computation by its unitary evolution. Unlike other representations, it is both universal – it can represent arbitrary computations – and unique: two computations are identical if and only if their matrices are equal. This makes it invaluable as a resource in circuit optimisation: unlike local circuit optimisation approaches whose success might vary depending on the circuit representation and the local optimisation landscape [73], circuits that are resynthesised from unitaries will always be unique.

However, unitary decomposition (as well as unitary computation itself) is challenging and

scales very poorly. Finding any decomposition has exponential cost in the average case and finding efficient decompositions is even harder.

Using results from linear algebra, one can find general unitary decomposition schemes that will break down arbitrary unitaries into product of unitaries expressible in the quantum circuit model [78]. Approaches using the Cosine-Sine decomposition [76], the Quantum Shanon decomposition [79] and the QR decomposition [80] have been proposed. While asymptotically efficient for almost all unitaries, such strategies typically generate fixed-sized circuits and fail to synthesise short circuits even when such circuits exist. This makes this approach inadequate for circuit resynthesis optimisation use cases, as well as for circuits beyond three qubits.

To find optimal or near-optimal circuits, search-based approaches have been developed. Exhaustive brute force search in circuit space has been performed in [66], finding T-depth optimal circuits for up to 3 qubits. More recently, more elaborate search-based approaches with pruning heuristics such as the A* algorithm have been successfully applied to this problem [67–69].

An alternative approach in [72] is to fix a circuit template and optimise continuous parameters to approximate the target unitary. This makes it possible to synthesise circuits with device constraints in mind and to trade off decomposition accuracy for shallower circuit depth and lower noise. Using regularisers, this approach can also optimise for sparse results that reduce gate count, especially when followed up by local optimisations.

Loke et al. have also proposed an approach merging reversible circuit and unitary matrix synthesis in [74]. They show that searching for decompositions $U = PU'Q$, where P and Q are reversible circuits can yield shorter circuits when using the Cosine-Sine decomposition for the unitaries U and U'.

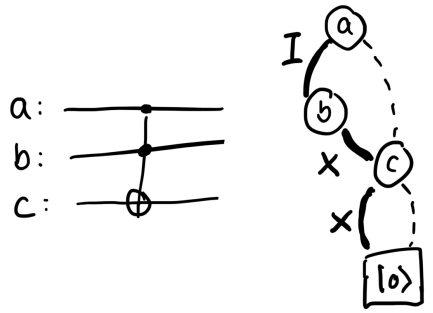
Such optimised unitary synthesis approaches yield efficient circuits but scale poorly beyond a handful of qubits. They can still be used on larger problems, however, by partitioning large circuits into smaller local subcircuits that can be resynthesised individually, yielding some of the best scalable circuit optimisation techniques developed to date [73].

iii. Other approaches

Beyond the well-established primitives discussed earlier, a series of other approaches have been presented.

Abdollahi and Pedram introduced Quantum Decision Diagrams (QDD) as an alternative primitive to quantum circuits [94]. QDDs are rooted directed acyclic graphs where vertices are annoted with a qubit and edges with quantum operations. The graph has a single terminal vertex, which has no outgoing edge. Every other vertex has exactly two outgoing edges, a 0-edge and a 1-edge. An input quantum state in the computational basis is then mapped to an output state by traversing the graph from the root to its terminal leaf, at each vertex following the 0 or 1-edge depending on the state of the annotated qubit. The authors present an efficient circuit synthesis algorithm from a QDD input, which for instance can be used to synthesise reversible circuits into quantum gates without first relying on classical decompositions.

An example adapted from the paper (Figure 7 in [94]), a QDD for a Toffoli gate, is reproduced here.



Starting at vertex a, the graph is traversed downward following the thick edge if the qubit associated with the vertex is set, or the dashed edge otherwise. Thick edges have an annotated quantum operation (in this case, a X or I

Pauli operation) that must be composed when traversed and applied to the $|0\rangle$ state of the terminal square vertex.

The paper gives a few more examples that can be represented as QDDs and more general computations beyond simple reversible circuits should be expressible as QDDs using other gates. However, no proof of universality is provided, leaving the general question open.

Another approach was presented in [96], where the Karnaugh Map notation from classical logic is adapted to the quantum case. The authors show that by trying to extend so-called bubbles in much the same way as in the classical case, one can obtain noteable improvements in circuit gate count. This approach also needs to be described and benchmarked more thoroughly in order to properly assess its potential.

Finally, Pauli gadgets are another set of quantum primitives that have been shown to be useful for quantum computation description and circuit simplification [95]. Pauli gadgets are given by unitaries of the form $\exp(-iP\alpha/2)$, where P is the tensor product of Pauli matrices I, X, Y, Z. This primitive appears naturally in the simulation of Hamiltonian evolution in quantum chemistry application and specialised circuit optimisation techniques were developed for such operations [95].

4. Discussion

This review has highlighted how most approaches have clustered around reversible circuits and the unitary matrix representation. Research work into optimisations in the reversible circuit framework has been especially fruitful over a long period of time. More recently, as classical computation became ever cheaper and the focus shifted towards obtaining maximal performance on simple quantum computations, more attention was given to search-based optimal or near-optimal synthesis of arbitrary quantum computations using the unitary representation.

Optimal matrix synthesis strategies seem parrticularly amenable to localised circuit optimisation through subcircuit resynthesis. However, especially as quantum systems scale, it will not solve the circuit design challenge, as it does not provide any higher-order abstraction beyond the limits of matrix synthesis capabilities, currently at around 5-6 qubits. Given the sizes of the unitaries involved, it also quickly becomes unmanageable as a user interface, arguably even before the computer is overwhelmed.

In principle, reversible circuit suffer from a similar exponential blow-up in size. However, some of the proposals reviewed can scale much better than unitaries. Decision Diagrams and ESOP-based representations scale with the number of terms in the expressions considered – in practice, most reversible circuits considered are polynomially-sized expressions that can be synthesised efficiently for large numbers of qubits. Changing from one canonical representation to another can be quite expensive, however.

As quantum computers scale beyond the current scales, reversible circuits will gain in relevance again – in the long term, we expect such reversible circuits to make up significant portions of fault tolerant algorithms. We believe that there is significant scope for further research in the area. Most research in the area has aged significantly: new research working with actual hardware models in mind is needed to develop efficient solutions specifically made for real quantum hardware.

An approach would be to decompose reversible circuits to quantum gates directly, optimising for quantum gate depth instead of restricting considerations to the classical framework. In that aim, relative phases may prove a fruitful avenue of work. Maslov showed that the circuit depth of reversible circuits can be significantly reduced if relative phase differences are allowed: toffoli gates can be implemented in just 3 CNOT gates instead of 6 CNOTs for the exact decomposition [109]. Whilst the use cases of such a phase shifted toffoli gate are restricted in general, phase shifts play nicely with reversible circuits, as they can be commuted through such permutations, creating new optimisation opportunities.

In that respect, the propopsed Qantum Decision Diagram framework proposed in [94] stands out as it can represent phase shifts in a natural way, as annotations on the graph edges. It might be that relative phase optimisations will be more natural to express in this language. Similarly, Quantum Karnaugh maps [96] also seem to encode relative phases in a more intuitive way than circuits.

Another area of potential within reversible circuits is the use and trade off of ancilla qubits for circuit depth. Multiple proposals have established that ancilla qubits can be used for reversible circuit synthesis [61, 93]. These are promising first steps towards an optimised ancilla management system that can make use of free qubits on spare qubits of the hardware. Further work could explore exactly this tradeoff between additional ancillas and reduced circuit depth.

Part II
Quantum circuits in additive Hilbert space

In the first part, we reviewed the existing primitives for the design of quantum computations. We established that proposals so far have clustered around a few approaches that as quantum hardware matures, will face significant limitations. New hardware architectures and increasingly complex application needs will demand better and faster optimisation capabilities, as well as more powerful abstractions for algorithm design.

In this second part of the paper, we would like to propose a new diagrammatic model for quantum computation that offers a different perspective on circuit optimisation and algorithm design, while drawing on the extensive research in optimisation for reversible circuit and unitary synthesis. We call such diagrams Additive Circuits (AC). The key idea is to replace the multiplicative monoidal product in the familiar circuit model – the tensor product – with an additive structure. For this reason and to avoid confusion, we will from now on refer to the standard circuit model as the multiplicative model of quantum computation.

We contend that ACs are both valuable as a educational and research tool to frame quantum computation in a novel and intuitive way, as it may be to express new circuit optimisation techniques used to improve quantum computation execution on hardware. We will give elements indicating that this model is suitable for NISQ machines and quantum compiler optimisations, as additive circuits can readily be converted to their multiplicative equivalent. In fact, by being less tied to hardware considerations than multiplicative circuits, the additive model opens the door for entirely new optimisation strategies. How this additional freedom can be used for compiler optimisations will be discussed in more details towards the end of this paper in section 7.

The ZX-calculus [9] illustrates perfectly how such alternative models both forge a better understanding of the underlying quantum theory and yield new avenues for circuit optimisation. It reformulates unique and unintuitive features of quantum computation as simple visual operations in a graphical language [7]. This educational resource has in turn allowed researchers to propose new ground-breaking circuit optimisation strategies [10, 112].

We start by discussing the assumptions and disadvantages of the multiplicative model in section 5. We then proceed to introducing additive circuits in section 6, its generators and its associated graphical language. The remaining sections discuss future avenues of work, in particular new compiler and optimisation techniques (section 7), approaches to mitigate scaling problems of the additive representation (section 8) and conclude with a discussion (section 9).

5. Disadvantages of multiplicative circuits

The multiplicative circuit model has established itself as by far the most popular representation for quantum computations. In its simplest definition, the circuit model is an elegant abstract description of the intricate experimental and physical details involved in the hardware implementation of different quantum computing architecture: qubits in the circuits correspond to physical two-dimensional state spaces, while gates represent Hamiltonian evolutions that these systems can be sujected to. The proximity to hardware primitives also makes the multiplicative language convenient to formulate hardware properties such as architecture constraints, gate sets and noise models.

However, there are also disadvantages inherent with using exclusively a multiplicative model, which we elaborate on in the rest of this section.

i. Exponential state space

Quantum theory postulates that the combined state space of multiple qubits together is given by the tensor product of the individual spaces. As a consequence, the dimension of the state space associated to a qubit system grows exponentially with the number of qubits. This also means that the computational space of a quantum circuit must grow exponentially with the size of the circuit. While this is a necessity for any scalable representation of quantum computations, that has drawbacks when formulating problems of finite size.

First of all, circuits might not be an ideal representation for computations that try to cover the state space densely. This could for example be relevant for NISQ-era approaches that are not as concerned with scalability as they are with using the quantum resources efficiently [113]. In the case of variational ansaetze, this is typically mitigated by increasing circuit depth as the number of qubits is increased; there is however very little theoretical understanding of how such high dimensional spaces are parametrised by variational circuits, and indeed in practice such approaches have proven hard to optimise successfully [2]. These issues are also of interest more broadly beyond small scale variational NISQ ansaetze in problems such as circuit encoding of classical data, where we expect to benefit from efficient encoding schemes [114].

Another limitation of the exponentially sized state space is in the representation and processing of data that might not live in a vector space with dimensionality that is a power of 2. Transformations in such spaces cannot be represented in the circuit framework and require additional padding to the next power of 2 before they can be decomposed into a circuit. On top of severely obfuscating user code, different padding strategies will require different information encodings and might lead to more or less efficient circuit decompositions, making this problem non-trivial.

ii. Circuit-unitary mismatch

As we have started to see in the previous paragraphs, circuits have an awkward relationship with the unitary matrix representation of quantum computation.

To see why the circuit model doesn't fit arbitrary use cases particularly well, it serves to have a look at such unitary matrix representations of quantum circuits. Figure 2 gives an example of a simple circuit and its corresponding unitary matrix. How long would it take you to find out that these are merely local operations – and that in fact the first qubit remains unchanged? The single parameter θ – local in the circuit model – becomes highly non-local in the unitary representation. As we compose such beautiful unitaries together and the number of non-local parameters increases, the associated optimisation problem becomes quickly challenging.

On top of this, adding an idle qubit will double the number of non-zero entries and quadruple the size of the matrix. You can see why even an optimiser might start to struggle once there are a few qubits and several such

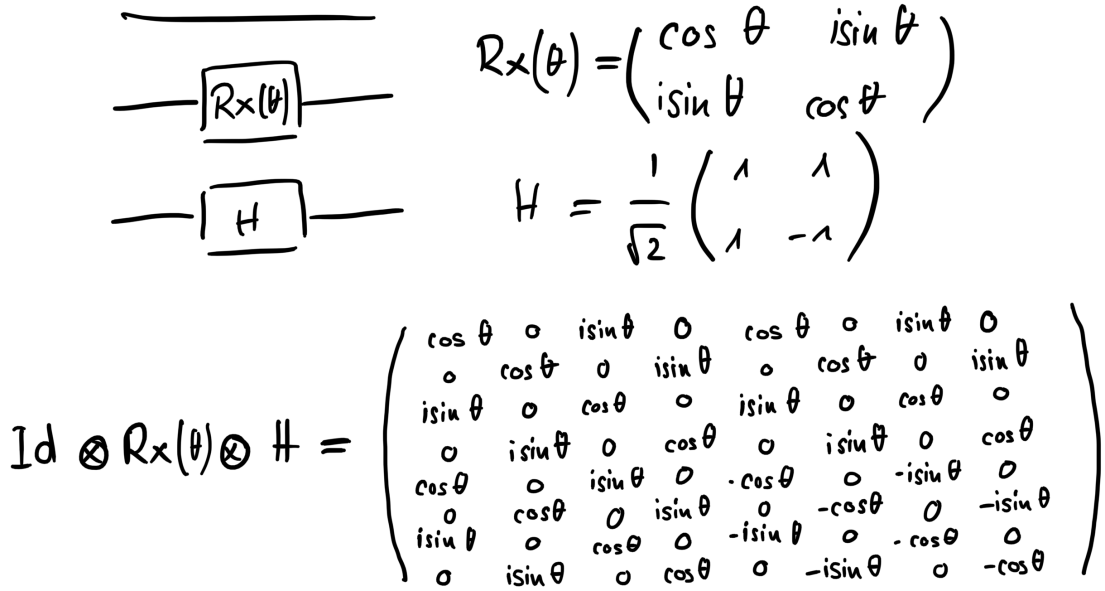


Figure 2: *This unitary looks bananas.*

matrices multiplied together – keeping in mind that such an optimiser in a NISQ context will usually only get to see obscure noisy measurement statistics, rather than the unitary.

For computational problems that have a natural map to quantum structures and quantum computation, this might not be an issue as there might be an evident interpretation of circuit primitives in the application domain. However, in many cases, we can only interpret circuits by the abstract transformation they implement overall, rather than the individual gates. Most variational approaches proposed [115, 116] fall in this category, as they optimise parameters based on general templates that cannot be application motivated. In such cases quantum circuits are not helpful to humans nor machines to understand what the computation is achieving. This might also make it difficult to express other familiar computational problems as quantum problems.

iii. Limited scope for optimisations

As devices have limited coherence times, quantum compilers play a crucial role in rewriting quantum computations to optimise execution performance. This will arguably become even more relevant over the course of the next few years as first quantum applications become viable and smart compilation will make the difference between a feasible and unfeasible circuit. Writing quantum operations in the low-level circuit model has for a long time been the only option to obtain quantum programmes executable on actual hardware[1]. In the future, however, the exact feature of the circuit model that has made it so useful, its closeness to the hardware, limits the optimisation potential of the compiler in several ways.

While the typical approach performs so-called "peephole" optimisations locally on the circuit to reduce computational complexity, the scope for improvements in this area are limited.

[1] Very rarely, one might want to define an operation directly by its unitary, which thanks to recent work and the ever-growing classical compute power available has become feasible up to 5-6 qubits.

As work on unitary synthesis has shown [68, 73], we can only hope to approach optimal solutions if we are able to perform non-local transformations to the circuit. For that, the circuit representation is actually a hindrance more than a useful tool: powerful non-local optimisation techniques such as unitary resynthesis [68], Pauli gadget synthesis [117] or depth reduction using ancilla qubits [61] cannot be performed until the circuit has been resynthesised into a non-local representation, in the hope of uncovering computational structure invisible in the circuit representation.

Beyond circuit optimisation, a core compiler function is to resolve hardware constraints so that a given quantum program can be executed, within the limitations of the device. The challenges of resynthesising quantum circuits into more helpful non-local descriptions also mean that we may struggle to compile for future hardware architectures. Most devices available today have broadly equivalent requirements, especially with regards to the available gate set [118–121]. This has meant that quantum compilers have been able to remap any quantum circuit between devices' native gate sets with minimal cost.

However, as new quantum technologies mature and qubit control improves, it is natural to expect that more specialised hardware primitives will become available. It might then become increasingly difficult to rebase to and make full use of these advanced gate sets from a circuit representation. As an example, ion trap architectures are typically able to perform entangling operations between three or more qubits at once [122–124]. Synthesising such operations from typcial circuit-level primitives is challenging. Even if these new gate types are accepted in the circuit input to the compiler, the burden would then fall on the user to make use of these unintuitive primitives and retargetting to other architectures would be impossible. The same problems would also arise if new architectures embraced higher-dimensional computational units, e.g. qutrits or qudits: our compilers would not be able to compile qubit circuits to such systems.

Finally, rigid quantum circuit descriptions disallow underspecified computations, which can be extremely valuable for compiler optimisations. In cases where outputs might be discarded or intermediate quantum computations are uncomputed later on, the user ought to be able to leave a level of flexibility to the compiler, instead of having to find themselves a unitary representation of the problem at hand.

6. Additive Circuits

We propose in this section a new additive model of quantum computation that addresses some of the issues discussed in the previous section. Like the ZX calculus [8, 9] and to a certain extent the standard circuit model itself, our new graphical language is rooted in the theory of symmetric monoidal categories, and more specifically PROPs. PROPs have a very rich and well-studied diagrammatic representation that we can leverage in our new model [125].

Crucially, however, we replace the tensor product structure that the ZX calculus and the multiplicative circuit model have inherited from the quantum postulates with an additive structure given by the direct sum of vector spaces. This makes our model a close parent to earlier work in graphical representations of boolean circuits and signal flow graphs [126–128]. The main difference is that we consider generators for the underlying symmetrical monoidal theory that are unitary linear operations. As far as we know, this has not been explored before.

Concretely, we represent each component of the state vector in the quantum Hilbert space as its own entity, and observe its transformation across time. For instance, a 3-qubit system would be viewed as $2^3 = 8$ individual entities, which we will call wires. Instead of considering the PROP object $A = \bigotimes_{i=1}^{3} Q$, where $Q = \mathbb{C}^2$ is the state space of a single qubit, we view it as the object $\tilde{A} = \bigoplus_{i=1}^{8} W$, where $W = \mathbb{C}$ is the one dimensional vector space of a single wire, an amplitude in the state vector.

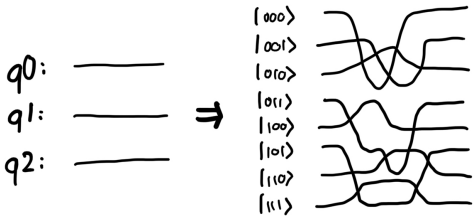
i. Diagrammatic representation

We use the well-developed diagrammatic language for PROPs [125] to represent additive circuits. The object $\tilde{A} = \bigoplus_{i=1}^{8} L$ is represented by 8 horizontal wires. Boxes are then appended on selected wires from left to right to represent function application and composition.

Unlike in circuits, wires are free to cross eachother as they like to connect to different gates. For that we introduce the wire swap operation τ

that corresponds to the permutation matrix that swaps two of its vector components. Applying a swap twice is again the identity – this is the symmetry of the monoidal product.

Composing such swaps on different wires, we can express arbitrary permutations. The precise swap order is immaterial within a given permutation; this allows us to write crossing wires in our diagrams without ambiguity:



By convention, what does matter is the order of the inputs and outputs in the additive circuit, as they define the mapping to the computational basis.

Note that such permutations correspond exactly to the reversible classical circuits that were studied in depth in the first part of this paper. CNOTs, Toffolis and NOT gates are all expressed by such permutations.

ii. Generators

We present a set of generators that make the additive model universal for quantum computing.

Phase shift

The phase shift acts on a single wire to add a phase to it: $|\psi\rangle \to e^{i\alpha} |\psi\rangle$. Note that quantum states cannot be distinguished up to global phase; the point of the phase shift is to introduce a phase difference relative to the other wires. We represent it in a diagram with a circle parametrised by α.

As a unitary, a phase shifter on a wire within a larger additive circuit corresponds to a diagonal operation, with ones everywhere except on the target wire. For instance when acting on a system with two wires, which would be written as a single qubit in the multiplicative model, a phase shift corresponds to a Rz rotation up to global phase:

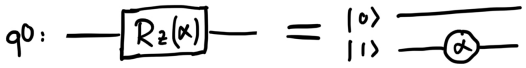


Figure 3 shows the same phase shift within a larger systems, in which it can be implemented as a phase gate controlled by the other qubits. This can alternatively be decomposed into a cascade of controlled Rz rotations, as can be seen in the figure.

Beam splitter

The second unitary generator acts on two wires, similar to the familiar beam splitter in optics: an input on the first wire such as $(1\ 0)^\top$ is transformed into $(t\ \sqrt{1-t^2})^\top$. The beam splitter can be parametrised with $\theta = \operatorname{acos} t$ and expressed as a $Ry(\theta)$ rotation:

$$Ry(\theta) = \exp\left(-iY\frac{\theta}{2}\right) = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & \sin\left(\frac{\theta}{2}\right) \\ -\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix},$$

Figure 3: *A single phase shift (left) translated as a multiplicative circuit (right).*

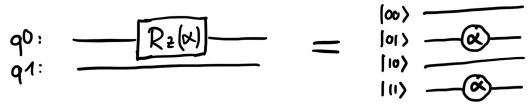
where $Y = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ is one of the Pauli matrices.

When the two wires that the beam splitter acts on are mapped to computational basis states that only differ in one qubit, i.e. their bitstrings have Hamming distance 1, the beam splitter can be implemented with a controlled $Ry(\theta)$ rotation in much the same way as the phase shift was implemented with a controlled phase gate. Otherwise, we can always apply a permutation first, implemented by a reversible classical circuit, such that the Hamming distance between the bitstrings is reduced to 1.

The phase shift and the beam splitter can be understood very intuitively: the first modulates the phase of a circuit wire, whilst the second modulates amplitude. While phase shifts can be performed on a single wire, the amplitude changes must use two different wires to preserve unitarity: we effectively reduce or increase amplitudes by mixing wires of different amplitudes.

iii. A universal model for quantum computation

Using the definitions of the phase shift and beam splitter, any Rz or Ry rotation in the multiplicative circuit model can be expressed in the additive model. Assume we would like to apply a rotation on qubit i, on a circuit with n qubits. Up to global phase, a Z rotation of angle θ can be expressed by adding θ-phase shifts to every other wire, that is to all wires that are mapped to a bitstring $|a_1 \dots a_{i-1} 1 a_{i+1} \dots a_n\rangle$ for some $a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_n \in \{0, 1\}$. Similarly, a Y rotation is obtained with beam splitters between bitstrings $|a_1 \dots a_{i-1} 0 a_{i+1} \dots a_n\rangle$ and $|a_1 \dots a_{i-1} 1 a_{i+1} \dots a_n\rangle$, for any such $a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_n \in \{0, 1\}$. For instance in the two qubit case, an $Rz(\theta)$ gate is obtained as follows (up to global phase):



Given that CNOTs are a reversible classical circuit and can be obtained with wire swaps in the additive model, we can thus express any circuit in the universal gate set $\{Ry, Rz, CNOT\}$ as an additive circuit. Figure 4 shows a translation from the standard circuit model to the additive circuit for the $\exp(-iZZ\theta/2)$ phase gadget.

iv. Illustrating phase and amplitude

We have described in the previous paragraphs how circuits of the additive model can be written as diagrams, using wires for each dimension of the underlying vector space and boxes for their unitary transformations.

Such a diagram is an alternative representation for the unitary U of the computation. In the case of circuits that are run on machines, however, only the image $U|0\rangle$ of the zero state is actually of interest. If we think in term of simulation of quantum devices, what we are interested in are the state vectors that evolve throughout the circuit starting from $|0\rangle$, rather than the full unitary. In this case, we can overlay additional information on top of additive circuits to display the state vector information.

We choose to represent the phase of each vector component with a colour gradient and

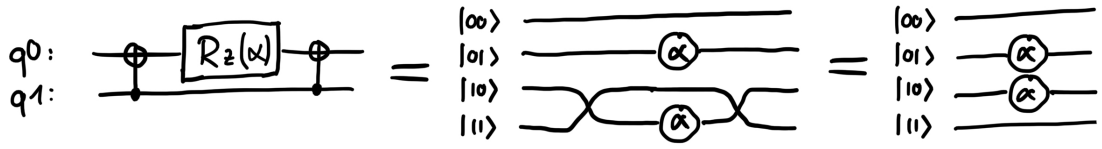
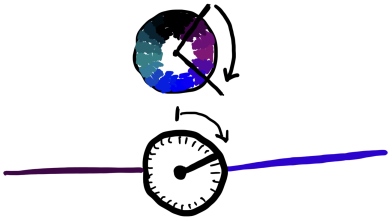


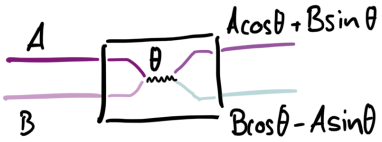
Figure 4: *A phase gadget implementing the* $\exp(-i\theta/2ZZ)$ *gate as an additive circuit.*

the amplitude with transparency. A phase shift can for example be pictured as follows

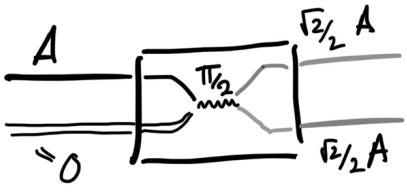


The colour wheel on top of the circuits represent the colour gradient from 0 phase to 2π.

Amplitude changes on the other hand can be observed when using beam splitters. For two wires of same phase but different amplitudes, we obtain



for an arbitrary mixing parameter θ. Note that here we drew the case where $Bcos\theta - Asin\theta < 0$, so that the second output incurs a phase flip, and hence a change in colour. For the special case were the second input has zero amplitude and the angle parameter $\theta = \pi/2$ is fixed



In this case, the beam splitter splits the first input wire into two wires of equal amplitude.

Building on fig. 4, an additive circuit starting in the $|0\rangle$ state might then look like the example in fig. 5.

7. Compiling Additive Circuits

Additive circuits will only ever be as useful as our ability to convert them into efficient circuits for execution on quantum devices. Excitingly, besides being very legible, additive circuits open a lot of doors for quantum optimisation.

i. Routing

The first opportunities for optimisation come from the difference in locality. In the multiplicative model operations are tied to logical qubits and therefore, up to placement, to hardware qubits on NISQ devices. In contrast, in the case of additive circuits, we can reroute every single state vector component to maximise performance given locality properties of the circuit and architecture constraints of the device.

Routing in the additive model also has the potential of being cheaper given its finer granuality than circuits: most traditional routing solutions involve exchanging qubits using swap gates that cost three CNOTs, whereas routing in the additive model can make use of reversible circuits composed of just single CNOTs. Finally, this finer graining will also mean that additive routing could make use of additional available ancillas in a way that is not possible in the multiplicative model, by mapping some amplitudes of the state vector to unused qubits.

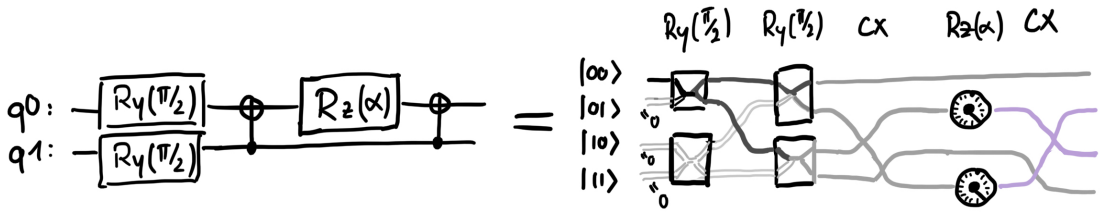


Figure 5: *A circuit (left) with its representation in the additive model, where paths have been coloured to illustrate the state vector (right).*

ii. Reversible circuits

Reversible circuits are one of the few classes of circuits that we are actually able to optimise well (see first part of the paper). Additive circuits give us the possibility to leverage such optimisations both by providing first class support for them in circuit design and construction as well as for internal use in the compiler, for instance for routing and more general strategies that might benefit from rearranging the vector components. Reversible circuits are also key to scaling quantum computations beyond small hand optimised circuits, as they form the core of most oracle-based and other large scale proposals for quantum algorithms [4, 129].

Not only are reversible circuits native in the additive model: they are totally implicit, as wires can be uncrossed and swapped without any change in semantics, as long as the input and output ordering are fixed. Thus in this model, CNOTs become mere wiring details that can be abstracted away and resynthesised only when and where necessary.

iii. Naturality of reversible circuits

Tying in with the previous point, wire swaps in the additive model are natural in the categorical sense: that is, one can *push through* all phase shifts and beam splitters past swaps so that, it would be possible to totally unentangle all wires, leaving only a layer of phase shits and beam splitters first, followed by a layer of wire permutations.

Furthermore, relative phase differences obtained by phase shifts are only relevant when beam splitters with another wire are performed; in all other cases, they can either be merged in with other phase shifts, pushed through permutations or discarded before measurements. This means that one optimisation strategy could involve removing any phase shifts from an additive circuit, keeping annotations of relative phase differences and reinserting phase shifts on one of the wires before beam splitters to restore the necessary phase differences. This also opens the door to further optimisations in reversible circuit synthesis, by synthesising such circuits up to relative phase. This could lead to significant gains in circuit depth, see [109] and the review discussion in section 4.

8. Scaling of the additive model

By far the biggest disadvantage that we have to contend with in this model of computation, by its very nature, is its exponential scaling. We try to address this in this section by discussing possible mitigation strategies.

The key will be to quickly build abstractions on top of basic generators, so that we can bundle large numbers of wires together. We could imagine *stacking* wires together and applying phase shifts and beam splitters, or more advanced primitives, in batches to such stacks. A single rotation in the multiplicative level would then correspond to some permutation of the wires in the additive model, followed by a single phase shift (or beam splitter) on the stacked wires.

The compositionality and scalability of our approach can be improved even further by

considering the multiplicative and additive monoidal products in a combined model. We would only use the additive model within the mulitplicative framework on subcircuits when useful, either for circuit design or for optimisation purposes. The interplay between both, given by distributivity laws, is very well understood at a categorical level [130] and could form the bedrock of the theory. In practice, we would select subcircuits in the multiplicative model and "explode" its n qubits into 2^n wires, which could then be optimised, rerouted or resynthesised as necessary.

In this way we can maintain the scalability of design, routing and optimisation procedures while still benefiting from the unique opportunities that the additive model can offer.

9. Discussion and future work

In this paper, we have introduced an additive model for quantum computation that addresses some of the conclusions that we drew from the litterature review in the first part. It provides a more abstract representation that will allow the user to think at a higher level of abstraction and could enable more powerful novel optimisation techniques. It makes use of the wealth of optimisation techniques developed for reversible circuits, and relative phase synthesis, as well as new optimisations such as simpler use of ancillas and more flexible routing. Finally, we are convinced that the additive model is an excellent tool for educational purposes, as the generators and overall theory are very simple to understand.

It highlights the exponential scaling of the computational state space instead of leaving it implicit as in the multiplicative model. This comes at a cost in its scaling performance, but could make it invaluable to visualise the power of quantum computation; it can for instance make the reason clear why large variational quantum circuit optimisation might fail. The exponential scaling can further be mitigated with the strategies detailed in section 8.

Unfortunately, we had to leave some motivations and descriptions at a very high level. We will elaborate in future work on some of the points discussed and tackle the issues one by one with more technical and thorough studies. Beyond the key scaling considerations that we have already addressed, there are several other open questions that need further work. First and foremost, efficient circuit synthesis approaches are needed to synthesise hardware-executable circuits out of additive representations. The naive approach – converting each additive primitive to a quantum gate – will be highly inefficient, as it will replace each primitive in an n qubit circuit with an n-controlled gate. This would in particular cause an exponential blow-up when converting a mulitplicative circuit into an additive one and back.

In order to synthesise more efficiently, we need to find maps of the additive wires onto the computational basis that simplify the quantum gates and reduce the number of controlled qubits. Stacking wires and generators as we proposed will also be part of the solution. Other related work in routing strategies and efficient reversible circuit synthesis will also be key in obtaining efficient circuits.

In parallel, the additive computation model should be formalised with precise categorical syntax and semantics. We have mentioned that the model is based on a PROP, which comes with a formal diagrammatic reasoning that we have used. We have also mentioned the links to other established theories such as the ZX-calculus [8, 9] and the graphical linear algebra [127, 128]. In further work, the relations and definitions should be made explicit. In particular the interaction of additive and multiplicative tensor product should be formalised in order to develop a compositional theory that combines both approaches.

A different avenue of work would be to study the convergence of variational optimisation algorithms in the additive model versus their conventional multiplicative counterpart. This could be another application area of additive circuits. We made the argument in section ii that since parameters affect the overall computation more locally in the additive model, it should perform better in variational

optimisation use cases. Different parametrisations and circuit templates would have to be explored to establish whether there is indeed an advantage in such use cases.

Additive circuit parametrisations could also provide a starting point for unitary synthesis in the additive model. Given that the additive circuit model is related to the graphical linear algebra, in which it well-known how to express matrices as diagrams [127], unitaries might be expressible quite simply in the additive model. This could be particularly interesting for instance for sparse matrices, for which there currently exists no dedicated synthesis methods.

Finally, the additive model provides an excellent opportunity to develop new intuitions on quantum computing for educational and research purposes. We ought to develop an interactive tool that allows users novice and experienced alike to design quantum circuits in the additive (or the combined additive-multiplicative) language. Such additive circuits could then be compiled into quantum circuits and run on simulators and actual devices.

We need scalable solutions that will allow humans to write circuits in the thousands of qubits, within a few years. The million dollar question is which framework will allow us to achieve this. We had better start looking for alternatives now, as I am sceptical that the circuit model on its own will be the solution.

References

[1] Michael A Nielsen and Isaac Chuang. *Quantum computation and quantum information*. 2002.

[2] Jarrod R. McClean et al. "Barren plateaus in quantum neural network training landscapes". In: *Nature Communications* 9.1 (Nov. 2018). DOI: 10.1038/s41467-018-07090-4.

[3] Peter W. Shor. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1484–1509. DOI: 10.1137/s0097539795293172.

[4] Lov K. Grover. "A Fast Quantum Mechanical Algorithm for Database Search". In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*. STOC '96. New York, NY, USA: Association for Computing Machinery, 1996, pp. 212–219. DOI: 10.1145/237814.237866.

[5] Philipp Schindler et al. "A quantum information processor with trapped ions". In: *New Journal of Physics* 15.12 (Dec. 2013), p. 123012. DOI: 10.1088/1367-2630/15/12/123012.

[6] T. P. Harty et al. "High-Fidelity Preparation, Gates, Memory, and Readout of a Trapped-Ion Quantum Bit". In: *Physical Review Letters* 113.22 (Nov. 2014). DOI: 10.1103/physrevlett.113.220501.

[7] Bob Coecke and Aleks Kissinger. "Picturing Quantum Processes". In: *Diagrammatic Representation and Inference*. Ed. by Peter Chapman et al. Cham: Springer International Publishing, 2018, pp. 28–31.

[8] Bob Coecke and Ross Duncan. "Interacting Quantum Observables". In: *Automata, Languages and Programming*. Springer Berlin Heidelberg, 2008, pp. 298–310. DOI: 10.1007/978-3-540-70583-3_25.

[9] John van de Wetering. *ZX-calculus for the working quantum computer scientist*. 2020. arXiv: 2012.13966 [quant-ph].

[10] Ross Duncan et al. *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*. Feb. 2019. arXiv: 1902.03178 [quant-ph].

[11] JabRef Development Team. *JabRef — An open-source, cross-platform citation and reference management software*. Version 5.1. 2021.

[12] D. Maslov, G.W. Dueck, and D.M. Miller. "Fredkin/Toffoli templates for reversible logic synthesis". In: *ICCAD-2003. International Conference on Computer Aided Design (IEEE Cat. No.03CH37486)*. Nov. 2003, pp. 256–261. DOI: 10.1109/ICCAD.2003.1257667.

[13] D. Maslov, D.M. Miller, and G.W. Dueck. "Templates for reversible circuit simplification". In: *PACRIM. 2005 IEEE Pacific Rim Conference on Communications, Computers and signal Processing, 2005.* Aug. 2005, pp. 609–612. DOI: 10.1109/PACRIM.2005.1517363.

[14] D Maslov, GW Dueck, and DM Miller. "Synthesis of Fredkin-Toffoli reversible networks". In: 13.6 (June 2005), pp. 765–769. DOI: 10.1109/TVLSI.2005.844284.

[15] D. Maslov, G.W. Dueck, and D.M. Miller. "Techniques for the synthesis of reversible Toffoli networks". In: *ACM Transactions on Design Automation of Electronic Systems* 12.4 (2007). DOI: 10.1145/1278349.1278355.

[16] Mathias Soeken et al. "Window Optimization of Reversible and Quantum Circuits". In: *Proceedings of the 13th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems*. IEEE Computer Soc, 2010, 341–345.

[17] Chander Chandak et al. "Analysis and Improvement of Transformation-Based Reversible Logic Synthesis". In: *2013 IEEE 43rd International Symposium on Multiple-Valued Logic*. May 2013, pp. 47–52. DOI: 10.1109/ISMVL.2013.14.

[18] H. Jayashree V, V. K. Agrawal, and Shishir N. Bharadwaj. "Post Synthesis Optimization of Reversible Logic Functions With Extended Template Matching". In: *2014 International Conference On Circuits, Communication, Control and Computing (I4C)*. IEEE, 2014, 368–371.

[19] M.J. Deshpande, H.V. Jayashree, and V.K. Agrawal. "Reversible circuit optimization using modified toffoli templates". In: IEEE, 2017, pp. 344–349. DOI: 10.1109/ICACCI.2017.8125864.

[20] Belayet Ali et al. "Quantum Cost Reduction of Reversible Circuits Using New Toffoli Decomposition Techniques". In: *2015 International Conference on Computational Science and Computational Intelligence (CSCI)*. Ed. by Arabnia, HR and Deligiannidis, L and Tran, QN. Amer Council on Sci & Educ. IEEE, 2015, 59–64. DOI: 10.1109/CSCI.2015.41.

[21] N.O. Scott and G.W. Dueck. "Pairwise decomposition of Toffoli gates in a quantum circuit". In: 2008, pp. 231–235. DOI: 10.1145/1366110.1366168.

[22] M. Arabzadeh, M. Saeedi, and M.S. Zamani. "Rule-based optimization of reversible circuits". In: 2010, pp. 849–854. DOI: 10.1109/ASPDAC.2010.5419684.

[23] K. Datta et al. "Exploiting negative control lines in the optimization of reversible circuits". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7948 LNCS (2013), pp. 209–220. DOI: 10.1007/978-3-642-38986-3_17.

[24] Md Zamilur Rahman and Jacqueline E. Rice. "Templates for Positive and Negative Control Toffoli Networks". In: *Reversible Computation, RC 2014*. Ed. by Yamashita, S and Minato, SI. Vol. 8507. Lecture Notes in Computer Science. Springer International Publishing AG, 2014, 125–136.

[25] K. Datta, I. Sengupta, and H. Rahaman. "A post-synthesis optimization technique for reversible circuits exploiting negative control lines". In: *IEEE Transactions on Computers* 64.4 (2015), pp. 1208–1214. DOI: 10.1109/TC.2014.2315641.

[26] P.M. Arpita et al. "Optimization of reversible circuits using triple-gate templates at quantum gate level". In: Institute of Electrical and Electronics Engi-

neers Inc., 2015, pp. 120–124. DOI: 10.1109/EDCAV.2015.7060551.

[27] N. Abdessaied et al. "Technology mapping of reversible circuits to Clifford+T quantum circuits". In: vol. 2016-July. IEEE Computer Society, 2016, pp. 150–155. DOI: 10.1109/ISMVL.2016.33.

[28] M. Gado and A. Younes. "Optimization of reversible circuits using toffoli decompositions with negative controls". In: *Symmetry* 13.6 (2021). DOI: 10.3390/sym13061025.

[29] M.M. Rahman, G.W. Dueck, and A. Banerjee. "Optimization of reversible circuits using reconfigured templates". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7165 LNCS (2012), pp. 43–53. DOI: 10.1007/978-3-642-29517-1_4.

[30] M. Soeken et al. "Optimizing the mapping of reversible circuits to four-valued quantum gate circuits". In: 2012, pp. 173–178. DOI: 10.1109/ISMVL.2012.64.

[31] Mehdi Saeedi, Robert Wille, and Rolf Drechsler. "Synthesis of quantum circuits for linear nearest neighbor architectures". In: 10.3 (June 2011), 355–377. DOI: 10.1007/s11128-010-0201-2.

[32] V.V. Shende et al. "Reversible logic circuit synthesis". In: *IEEE/ACM International Conference on Computer Aided Design, 2002. ICCAD 2002.* Nov. 2002, pp. 353–360. DOI: 10.1109/ICCAD.2002.1167558.

[33] D.K. Kole et al. "Optimal reversible logic circuit synthesis based on a hybrid DFS-BFS technique". In: 2010, pp. 208–212. DOI: 10.1109/ISED.2010.47.

[34] O. Golubitsky, S.M. Falconer, and D. Maslov. "Synthesis of the optimal 4-bit reversible circuits". In: 2010, pp. 653–656. DOI: 10.1145/1837274.1837440.

[35] M. Szyprowski and P. Kerntopf. "Optimal 4-bit reversible mixed-polarity toffoli circuits". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7581 LNCS (2013), pp. 138–151. DOI: 10.1007/978-3-642-36315-3_11.

[36] Z. Li et al. "A synthesis algorithm for 4-bit reversible logic circuits with minimum quantum cost". In: *ACM Journal on Emerging Technologies in Computing Systems* 11.3 (2014). DOI: 10.1145/2629542.

[37] X. Wang, L. Jiao, and Y. Li. "A hybrid algorithm for reversible toffoli circuits synthesis". In: *Communications in Computer and Information Science* 463 (2014). Ed. by Li K. et al., pp. 227–239. DOI: 10.1007/978-3-662-45286-8_24.

[38] Yi-Tzu Lo et al. "A Novel Algorithm for Reversible Circuit Optimization". In: *IEEE International Conference on Systems, Man, and Cybernetics*. IEEE COMPUTER SOC, 2015, 2867–2872. DOI: 10.1109/SMC.2015.499.

[39] K. Datta et al. "Exact Synthesis of Reversible Circuits Using A* Algorithm". In: *Journal of The Institution of Engineers (India): Series B* 96.2 (2015), pp. 121–130. DOI: 10.1007/s40031-014-0133-9.

[40] M Lukac et al. "Evolutionary approach to Quantum and Reversible Circuits synthesis". In: 20.3-4 (Dec. 2003), 361–417. DOI: 10.1023/B:AIRE.0000006605.86111.79.

[41] P. Manna et al. "Reversible logic circuit synthesis using genetic algorithm and particle swarm optimization". In: 2012, pp. 246–250. DOI: 10.1109/ISED.2012.71.

[42] T.N. Sasamal, A.K. Singh, and A. Mohan. "Reversible Logic Circuit Synthesis and Optimization Using Adaptive Genetic Algorithm". In: ed. by Mathew J. Singh A.K. Vol. 70. Elsevier B.V., 2015,

pp. 407–413. DOI: 10.1016/j.procs.2015.10.054.

[43] Mustapha Yusuf Abubakar et al. "New universal gate library for synthesizing reversible logic circuit using genetic programming". In: *2016 3rd International Conference on Computer and Information Sciences (ICCOINS)*. Aug. 2016, pp. 316–321. DOI: 10.1109/ICCOINS.2016.7783234.

[44] M. Mohammadi and M. Eshghi. "Behavioral description of quantum V and V+ gates to design quantum logic circuits". In: 2008. DOI: 10.1109/SSD.2008.4632850.

[45] A. Agrawal and N.K. Jha. "Synthesis of reversible logic". In: *Proceedings Design, Automation and Test in Europe Conference and Exhibition*. Vol. 2. Feb. 2004, 1384–1385 Vol.2. DOI: 10.1109/DATE.2004.1269099.

[46] Zhijin Guan et al. "Reversible Synthesis with Minimum logic function". In: *2006 International Conference on Computational Intelligence and Security*. Vol. 2. Nov. 2006, pp. 968–971. DOI: 10.1109/ICCIAS.2006.295405.

[47] K. Fazel, M. A. Thornton, and J. E. Rice. "ESOP-based Toffoli Gate Cascade Generation". In: *2007 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*. Aug. 2007, pp. 206–209. DOI: 10.1109/PACRIM.2007.4313212.

[48] L. Tran et al. "Synthesis of reversible circuits based on EXORs of products of EXORs". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8911 (2014), pp. 111–128. DOI: 10.1007/978-3-662-45711-5_7.

[49] Chandan Bandyopadhyay, Hafizur Rahaman, and Rolf Drechsler. "A Cube Pairing Approach for Synthesis of ESOP-Based Reversible Circuit". In: *2014 IEEE 44th International Symposium on Multiple-Valued Logic*. May 2014, pp. 109–114. DOI: 10.1109/ISMVL.2014.27.

[50] N.A.B. Adnan, K. Kushida, and S. Yamashita. "A pre-optimization technique to generate initial reversible circuits with low quantum cost". In: vol. 2016-July. Institute of Electrical and Electronics Engineers Inc., 2016, pp. 2298–2301. DOI: 10.1109/ISCAS.2016.7539043.

[51] S.P. Parlapalli, C. Vudadha, and M.B. Srinivas. "An ESOP based cube decomposition technique for reversible circuits". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 10301 LNCS (2017). Ed. by Phillips I. Rahaman H., pp. 127–140. DOI: 10.1007/978-3-319-59936-6_10.

[52] J. Jegier and P. Kerntopf. "PPRM-based approach to synthesis of reversible functions". In: ed. by Romaniuk R.S. Linczuk M. Vol. 10445. SPIE, 2017. DOI: 10.1117/12.2280943.

[53] Anirban Basak et al. "Cost Optimization Technique for Quantum Circuits". In: 58.9 (Sept. 2019), 3158–3179. DOI: 10.1007/s10773-019-04192-7.

[54] Y. Zheng and C. Huang. "A novel toffoli network synthesis algorithm for reversible logic". In: 2009, pp. 739–744. DOI: 10.1109/ASPDAC.2009.4796568.

[55] S. Wan, H. Chen, and R. Cao. "A novel transformation-based algorithm for reversible logic synthesis". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 5821 LNCS (2009), pp. 70–81. DOI: 10.1007/978-3-642-04843-2_9.

[56] O. Susam and M. Altun. "An efficient algorithm to synthesize quantum circuits and optimization". In: Institute of Electrical and Electronics Engineers Inc.,

2014, pp. 570–573. DOI: 10.1109/ICECS.2014.7050049.

[57] Ö. Susam and M. Altun. "Fast synthesis of reversible circuits using a sorting algorithm and optimization". In: *Journal of Multiple-Valued Logic and Soft Computing* 29.1-2 (2017), pp. 1–23.

[58] Md. Mazder Rahman et al. "Integrated Synthesis of Linear Nearest Neighbor Ancilla-Free MCT Circuits". In: *2016 IEEE 46th International Symposium on Multiple-Valued Logic (ISMVL)*. May 2016, pp. 144–149. DOI: 10.1109/ISMVL.2016.54.

[59] R. Wille and R. Drechsler. "Effect of BDD Optimization on Synthesis of Reversible and Quantum Logic". In: *Electronic Notes in Theoretical Computer Science* 253.6 (2010), pp. 57–70. DOI: 10.1016/j.entcs.2010.02.006.

[60] Y. Pang et al. "Positive Davio-based synthesis algorithm for reversible logic". In: 2011, pp. 212–218. DOI: 10.1109/ICCD.2011.6081399.

[61] S. Stojković et al. "Reversible Circuits Synthesis from Functional Decision Diagrams by using Node Dependency Matrices". In: *Journal of Circuits, Systems and Computers* 29.5 (2020). DOI: 10.1142/S0218126620500796.

[62] S. Banerjee et al. "Toffoli netlist based synthesis of four variable reversible functions". In: ed. by Piuri V. et al. Vol. 2017-December. Institute of Electrical and Electronics Engineers Inc., 2017, pp. 315–320. DOI: 10.1109/ICRCICN.2017.8234527.

[63] S. Thakral and D. Bansal. "Improved ant colony optimization for quantum cost reduction". In: *Bulletin of Electrical Engineering and Informatics* 9.4 (2020), pp. 1525–1532. DOI: 10.11591/eei.v9i4.1657.

[64] K. Datta et al. "A cycle based reversible logic synthesis approach". In: 2013, pp. 316–319. DOI: 10.1109/ICACC.2013.67.

[65] J. Jung and I.-C. Choi. "A multi-commodity network model for optimal quantum reversible circuit synthesis". In: *PLoS ONE* 16.6 June (2021). DOI: 10.1371/journal.pone.0253140.

[66] M. Amy et al. "A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 32.6 (2013), pp. 818–830. DOI: 10.1109/TCAD.2013.2244643.

[67] Marc G. Davis et al. "Towards Optimal Topology Aware Quantum Circuit Synthesis". In: *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*. Oct. 2020, pp. 223–234. DOI: 10.1109/QCE49297.2020.00036.

[68] Ethan Smith et al. *LEAP: Scaling Numerical Optimization Based Synthesis Using an Incremental Approach*. arXiv: 2106.11246 [quant-ph].

[69] Vlad Gheorghiu, Michele Mosca, and Priyanka Mukhopadhyay. *A quasi-polynomial time heuristic algorithm for synthesizing T-depth optimal circuits*. arXiv: 2101.03142 [quant-ph].

[70] A. Daskin and S. Kais. "Decomposition of unitary matrices for finding quantum circuits: Application to molecular Hamiltonians". In: *Journal of Chemical Physics* 134.14 (2011). DOI: 10.1063/1.3575402.

[71] E.A. Martinez et al. "Compiling quantum algorithms for architectures with multi-qubit gates". In: *New Journal of Physics* 18.6 (2016). DOI: 10.1088/1367-2630/18/6/063029.

[72] Liam Madden and Andrea Simonetto. *Best Approximate Quantum Compiling Problems*. arXiv: 2106.05649 [quant-ph].

[73] Xin-Chuan Wu et al. *QGo: Scalable Quantum Circuit Optimization Using Automated Synthesis*. arXiv: 2012.09835 [quant-ph].

[74] T. Loke, J. B. Wang, and Y. H. Chen. "OptQC: An optimized parallel quantum compiler". In: 185.12 (Dec. 2014), 3307–3316. DOI: 10.1016/j.cpc.2014.07.022.

[75] SS Bullock and IL Markov. "An arbitrary two-qubit computation in 23 elementary gates or less". In: *40th Design Autoation Conference, Proceedings 2003*. Design Automation Conference DAC. Assoc Computing Machinery, 2003, 324–329.

[76] M. Möttönen et al. "Quantum circuits for general multiqubit gates". In: *Physical Review Letters* 93.13 (2004), pp. 130502-1-130502–4. DOI: 10.1103/PhysRevLett.93.130502.

[77] M. Saeedi et al. "Block-based quantum-logic synthesis". In: *Quantum Information and Computation* 11.3-4 (2011), pp. 262–277.

[78] Raban Iten et al. *Introduction to UniversalQCompiler*. arXiv: 1904.01072 [quant-ph].

[79] A. M. Krol et al. *Efficient decomposition of unitary matrices in quantum circuit compilers*. arXiv: 2101.02993 [quant-ph].

[80] M. Sedlák and M. Plesch. "Towards optimization of quantum circuits". In: *Central European Journal of Physics* 6.1 (2008), pp. 128–134. DOI: 10.2478/s11534-008-0039-8.

[81] Xiaoming Sun et al. *Asymptotically Optimal Circuit Depth for Quantum State Preparation and General Unitary Synthesis*. arXiv: 2108.06150 [quant-ph].

[82] Yihui Quek and Patrick Rebentrost. *Fast algorithm for quantum polar decomposition, pretty-good measurements, and the Procrustes problem*. arXiv: 2106.07634 [quant-ph].

[83] V.V. Shende, I.L. Markov, and S.S. Bullock. "Smaller two-qubit circuits for quantum communication and computation". In: ed. by Figueras J. Gielen G. Vol. 2. 2004, pp. 980–985. DOI: 10.1109/DATE.2004.1269020.

[84] D. Maslov et al. "Quantum circuit simplification using templates". In: *Design, Automation and Test in Europe*. Mar. 2005, 1208–1213 Vol. 2. DOI: 10.1109/DATE.2005.249.

[85] D. Maslov et al. "Quantum circuit simplification and level compaction". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 27.3 (2008), pp. 436–444. DOI: 10.1109/TCAD.2007.911334.

[86] H.R. Bhagyalakshmi and M.K. Venkatesha. "Toffoli cascade synthesis of an optimized two-bit comparator". In: *Lecture Notes in Electrical Engineering* 248 LNEE (2014), pp. 779–787. DOI: 10.1007/978-81-322-1157-0_79.

[87] A. Banerjee, A. Pathak, and G.W. Dueck. "Minimal designs of reversible sequential elements". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8507 LNCS (2014), pp. 137–148. DOI: 10.1007/978-3-319-08494-7_11.

[88] H. V. Jayashree and V. K. Agrawal. "Design of Ancilla Invariant and Quantum Cost efficient reversible arithmetic components". In: *2017 International Conference on Innovations in Electronics, Signal Processing and Communication (IESC)*. Apr. 2017, pp. 22–27. DOI: 10.1109/IESPC.2017.8071858.

[89] A. Chaudhuri et al. "A novel reversible two's complement gate (TCG) and its quantum mapping". In: ed. by Sarkar A. Nandi S. Institute of Electrical and Electronics Engineers Inc., 2017, pp. 252–256. DOI: 10.1109/DEVIC.2017.8073946.

[90] A. Shafaei, M. Saeedi, and M. Pedram. "Reversible Logic synthesis of K-input, M-output lookup tables". In: Institute of Electrical and Electronics Engineers Inc., 2013, pp. 1235–1240. DOI: 10.7873/date.2013.256.

[91] Katherine L. Brown et al. "Reducing the number of ancilla qubits and the gate count required for creating large controlled operations". In: 14.3 (Mar. 2015), 891–899. DOI: 10.1007/s11128-014-0900-1.

[92] P. Niemann, A. Gupta, and R. Drechsler. "T-depth Optimization for Fault-Tolerant Quantum Circuits". In: vol. 2019-May. IEEE Computer Society, 2019, pp. 108–113. DOI: 10.1109/ISMVL.2019.00027.

[93] Yongshan Ding et al. "SQUARE: Strategic Quantum Ancilla Reuse for Modular Quantum Programs via Cost-Effective Uncomputation". In: *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA 2020)*. IEEE Computer Soc, 2020, 570–583. DOI: 10.1109/ISCA45697.2020.00054.

[94] A. Abdollahi and M. Pedram. "Analysis and synthesis of quantum circuits by using quantum decision diagrams". In: vol. 1. Institute of Electrical and Electronics Engineers Inc., 2006. DOI: 10.1109/date.2006.244176.

[95] A. Cowtan et al. "Phase gadget synthesis for shallow circuits". In: ed. by Leifer M. Coecke B. Vol. 318. Open Publishing Association, 2020, pp. 213–228. DOI: 10.4204/EPTCS.318.13.

[96] J.-H. Bae et al. "Quantum circuit optimization using quantum Karnaugh map". In: *Scientific Reports* 10.1 (2020). DOI: 10.1038/s41598-020-72469-7.

[97] Reza Haghshenas. "Optimization schemes for unitary tensor-network circuit". In: 3.2 (May 2021). DOI: 10.1103/PhysRevResearch.3.023148.

[98] G. Meuli, M. Soeken, and G. De Micheli. "SAT-based CNOT, T quantum circuit synthesis". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11106 LNCS (2018). Ed. by Kari J. Ulidowski I., pp. 175–188. DOI: 10.1007/978-3-319-99498-7_12.

[99] M. Houshmand et al. "Quantum circuit synthesis targeting to improve one-way quantum computation pattern cost metrics". In: *ACM Journal on Emerging Technologies in Computing Systems* 13.4 (2017). DOI: 10.1145/3064834.

[100] P. Mercy Nesa Rani et al. "Improved Decomposition of Multiple-Control Ternary Toffoli Gates Using Muthukrishnan-Stroud Quantum Gates". In: *Reversible Computation RC 2017*. Ed. by Phillips, I and Rahaman, H. Vol. 10301. Lecture Notes in Computer Science. Springer International Publishing AG, 2017, 202–213. DOI: 10.1007/978-3-319-59936-6_16.

[101] Amit Saha et al. *Asymptotically Improved Grover's Algorithm in any Dimensional Quantum System with Novel Decomposed n-qudit Toffoli Gate*. arXiv: 2012.04447 [quant-ph].

[102] Earl T. Campbell and Mark Howard. "Unified framework for magic state distillation and multiqubit gate synthesis with reduced resource cost". In: *Physical Review A* 95.2 (Feb. 2017). DOI: 10.1103/physreva.95.022316.

[103] Vadym Kliuchnikov, Dmitri Maslov, and Michele Mosca. "Practical Approximation of Single-Qubit Unitaries by Single-Qubit Quantum Clifford and T Circuits". In: 65.1 (Jan. 2016), pp. 161–172. DOI: 10.1109/TC.2015.2409842.

[104] Gary J. Mooney, Charles D. Hill, and Lloyd C. L. Hollenberg. "Cost-optimal single-qubit gate synthesis in the Clifford hierarchy". In: *Quantum* 5 (Feb. 2021), p. 396. DOI: 10.22331/q-2021-02-15-396.

[105] M. Zomorodi-Moghadam, M.-A. Taherkhani, and K. Navi. "Synthesis and optimization by quantum circuit description language". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8911 (2014), pp. 74–91. DOI: 10.1007/978-3-662-45711-5_5.

[106] Mehdi Saeedi and Igor L. Markov. "Synthesis and Optimization of Reversible Circuits-A Survey". In: 45.2 (Feb. 2013). DOI: 10.1145/2431211.2431220.

[107] Adriano Barenco et al. "Elementary gates for quantum computation". In: *Physical Review A* 52.5 (Nov. 1995), pp. 3457–3467. DOI: 10.1103/physreva.52.3457.

[108] Vivek V. Shende and Igor L. Markov. *On the CNOT-cost of TOFFOLI gates*. 2008. arXiv: 0803.2316 [quant-ph].

[109] Dmitri Maslov. "Advantages of using relative-phase Toffoli gates with an application to multiple control Toffoli optimization". In: *Physical Review A - Atomic, Molecular, and Optical Physics* 93.2, 022311 (Feb. 2016), p. 022311. DOI: 10.1103/PhysRevA.93.022311.

[110] Jonathan M. Baker et al. *Decomposing Quantum Generalized Toffoli with an Arbitrary Number of Ancilla*. 2019. arXiv: 1904.01671 [quant-ph].

[111] Philipp Niemann, Anshu Gupta, and Rolf Drechsler. "T-depth Optimization for Fault-Tolerant Quantum Circuits". In: *2019 IEEE 49th International Symposium on Multiple-valued Logic (ISMVL)*. IEEE, 2019, 108–113. DOI: 10.1109/ISMVL.2019.00027.

[112] Aleks Kissinger and Arianne Meijer-van de Griend. *CNOT circuit extraction for topologically-constrained quantum memories*. 2019. arXiv: 1904.00633 [quant-ph].

[113] Adrián Pérez-Salinas et al. "Data re-uploading for a universal quantum classifier". In: *Quantum* 4 (Feb. 2020), p. 226. DOI: 10.22331/q-2020-02-06-226.

[114] Maria Schuld, Ryan Sweke, and Johannes Jakob Meyer. "Effect of data encoding on the expressive power of variational quantum-machine-learning models". In: *Phys. Rev. A* 103 (Mar. 2021), p. 032430. DOI: 10.1103/PhysRevA.103.032430.

[115] David Amaro et al. *Filtering variational quantum algorithms for combinatorial optimization*. 2021. arXiv: 2106.10055 [quant-ph].

[116] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. *A Quantum Approximate Optimization Algorithm*. 2014. arXiv: 1411.4028 [quant-ph].

[117] Alexander Cowtan, Will Simmons, and Ross Duncan. *A Generic Compilation Strategy for the Unitary Coupled Cluster Ansatz*. 2020. arXiv: 2007.10515 [quant-ph].

[118] Petar Jurcevic et al. "Demonstration of quantum volume 64 on a superconducting quantum computing system". In: *Quantum Science and Technology* 6.2 (Mar. 2021), p. 025020. DOI: 10.1088/2058-9565/abe519.

[119] J. M. Pino et al. "Demonstration of the trapped-ion quantum CCD computer architecture". In: *Nature* 592.7853 (Apr. 2021), pp. 209–213. DOI: 10.1038/s41586-021-03318-4.

[120] J. M. Arrazola et al. "Quantum circuits with many photons on a programmable nanophotonic chip". In: *Nature* 591.7848 (Mar. 2021), pp. 54–60. DOI: 10.1038/s41586-021-03202-1.

[121] Frank Arute et al. "Quantum supremacy using a programmable superconducting processor". In: *Nature* 574.7779 (Oct. 2019), pp. 505–510. DOI: 10.1038/s41586-019-1666-5.

[122] S. Debnath et al. "Demonstration of a small programmable quantum computer with atomic qubits". In: *Nature* 536.7614 (Aug. 2016), pp. 63–66. DOI: 10.1038/nature18648.

[123] C. Figgatt et al. "Parallel entangling operations on a universal ion-trap quantum computer". In: *Nature* 572.7769 (July 2019), pp. 368–372. DOI: 10.1038/s41586-019-1427-5.

[124] Nikodem Grzesiak et al. "Efficient arbitrary simultaneously entangling gates on a trapped-ion quantum computer". In: *Nature Communications* 11.1 (June 2020). DOI: 10.1038/s41467-020-16790-9.

[125] John C. Baez, Brandon Coya, and Franciscus Rebro. "Props in Network Theory". In: (July 2017). arXiv: 1707.08321 [math.CT].

[126] Yves Lafont. "Towards an algebraic theory of Boolean circuits". In: *Journal of Pure and Applied Algebra* 184.2-3 (Nov. 2003), pp. 257–310. DOI: 10.1016/s0022-4049(03)00069-0.

[127] Filippo Bonchi, Pawel Sobocinski, and Fabio Zanasi. "Full Abstraction for Signal Flow Graphs". In: *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. ACM, Jan. 2015. DOI: 10.1145/2676726.2676993.

[128] Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. "Interacting Hopf algebras". In: *Journal of Pure and Applied Algebra* 221.1 (Jan. 2017), pp. 144–184. DOI: 10.1016/j.jpaa.2016.06.002.

[129] David Deutsch and Richard Jozsa. "Rapid solution of problems by quantum computation". In: *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439.1907 (Dec. 1992), pp. 553–558. DOI: 10.1098/rspa.1992.0167.

[130] Stephen Lack. "Composing props". In: *Theory and Applications of Categories* 13.9 (2004), pp. 147–163.