Optimizing Quantum Circuits using Algebraic Expressions

Varun Puram, Krishnageetha Karuppasamy, Johnson P Thomas
Oklahoma State University
Stillwater, Oklahoma. 74075, USA
vpuram@okstate.edu,kkarupp@okstate.edu, jpt@cs.okstate.edu

Abstract. Optimizing quantum circuits and reducing errors plays a crucial role in quantum circuit computation. Every quantum circuit can be represented using algebraic expressions, we propose an approach that directly derives algebraic expressions, ensuring that parallelism is maximized, that is, the number of circuit slices is minimized, and secondly, the computation required for obtaining the desired algebraic expression is reduced. This results in quantum circuits that are more efficient in space and computation time. The simplification of algebraic expressions offers methods to streamline optimized circuits, reducing the number of gates and depth. This reduction is aimed at minimizing the overall complexity of the expressions, resulting in more efficient quantum computations. In this paper, we also show through simulations that the optimized circuit will have less errors when compared to original circuits.

Keywords: Quantum Computing, Algebraic expressions, Quantum circuit optimization and Quantum error correction.

1 Introduction

Although Quantum computing algorithms have been proven to be much faster than classical algorithms, a major obstacle to its successful realization is high error or noise rates. Due to this noise, qubits are rotated by the wrong amount that cause the final output state to not be the final correct state that is expected [7]. Unlike classical circuit optimization, quantum circuit optimization not only improves the execution time but also helps to reduce error. This is because quantum noises are linearly dependent on the size of the circuits [8]. As a quantum circuit becomes larger and more complex, they involve a higher number of qubits and quantum gates. Larger circuits tend to require longer coherence times to complete the computations successfully. Qubit coherence time is finite. Hence, the probability of errors occurring during quantum operations increases. An error in an early gate (a gate that does the computations early in execution of an algorithm) can affect the state of qubits as they propagate through subsequent gates. Thus, by reducing the circuit size the errors can also be mitigated.

Quantum algorithms are implemented as quantum circuits, which consist of quantum gates. Quantum Circuit Optimization is the process of finding a more efficient and streamlined way to represent a given quantum circuit or algorithm. This is achieved by

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63778-0_19

reducing the number of gates used and shortening the overall depth of the circuit. The optimized circuit should implement the same algorithm as the original circuit and hence yield the exact same results as the original circuit while operating more efficiently in terms of errors and execution time.

There are several ways to represent quantum circuits such as Quantum Assembly Language (QASM) [10], Quantum Gate Matrices, Directed Acyclic Graphs (DAGs) [1]. Every representation has its own advantages and disadvantages. One common technique for optimization involves representing a quantum circuit as a Directed Acyclic Graph (DAG) with gate commuting properties. This approach allows for the cancellation of gates and the merging of gates to simplify the quantum circuit. [1-5, 9] proposes various optimization and pattern matching techniques using DAG.

In this paper we propose an algorithm to obtain efficient algebraic expressions for quantum circuits. Algebraic expressions ensure that the algorithm itself does not change because of optimization. Using algebraic expressions, quantum circuits can be analyzed and optimized. Algebraic simplification leads to less complex logic expressions which results in reduced circuit complexity. This leads to improved performance, lower power consumption and fewer errors. Simulations show that the resultant optimized circuit has less errors than the original circuit.

2 Proposed Approach

Quantum algorithms are expressed using quantum circuits, which are sequences of quantum gates. These gates perform quantum computations. Each quantum gate can be mathematically represented by a unitary matrix, which describes the transformation as it applies to quantum states. A quantum circuit can also be viewed as a composite unitary matrix. If there is a n-qubit quantum circuit, the corresponding unitary matrix to the circuit will have a dimension of $2^n \times 2^n$. At each depth or layer in the quantum circuit, a quantum gate acts on its respective qubit. If the quantum circuit has n depth of layers (or "slices") of parallel computations, then there will be n slices present. Multiplication of the slices result in composite unitary matrix of the quantum circuit.

When a quantum circuit contains single-qubit gates, they are represented using their respective matrices. When the circuit includes multi-qubit gates, these quantum matrices are decomposed into elementary unitary matrices. An efficient approach for this decomposition was developed by Hutsell et. al. [6]. They proposed the use of two 2×2 matrices, D_0 and D_1, as a means of decomposing larger unitary matrices into smaller 2×2 matrices. This decomposition allows for a more efficient representation of the algebraic expressions involved.

$$D_0 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad D_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \quad \{D_0 + D_1 = \mathrm{I}\}$$

where I is the identity matrix. We utilize this work [6] as a foundation for our proposed new approach.

We use Hutsell's work [6] to address the general case where a slice of a quantum circuit contains multiple target qubits acting on a control qubit or multiple multi-qubit gates. Figure 2a provides an illustration. Using the approach proposed in [6], the given

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63778-0_19

slice is broken down into further slices to achieve the desired algebraic expression, as demonstrated in Figure 1b. A slice refers to a layer of parallel computation.

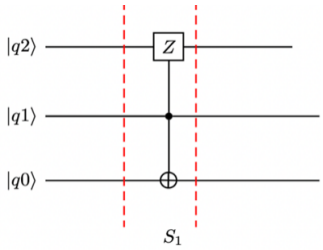

Figure 1a: Circuit with the slice which has 2 targets (Z and Not) with only 1 respective control qubit (|q1⟩)

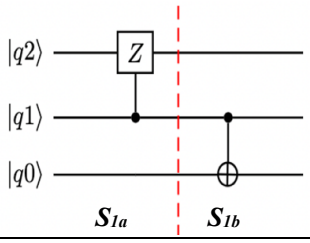

Figure 1b: Slice S1 is divided into two slices S1a and S1b to get algebraic expressions.

The disadvantage with this approach is when the number of target qubit exceeds the number of control qubits, the circuit will decompose into many slices which will result in an inefficient algebraic expression, resulting in a less optimized algebraic expression which leads to a less optimized circuit. Our proposed work introduces a novel method to handle situations where the number of target qubits exceeds the number of control qubits. Figure 2a provides an example. To get an algebraic expression for circuit S_1 the previous method [6] needs slicing into S_{1a} and S_{1b} (Figure 1b). The disadvantage of slicing is that it increases the depth of the circuit (Figure 1b). This means more slices which results in more inefficiency and errors. In quantum circuit computation, each slice of n qubits is represented by $2^n \times 2^n$ Unitary matrix, Although the final matrix remains unchanged, obtaining the final matrix involves more matrix multiplications as the number of slices increases. There is a computation for each slice followed by a computation to merge the results from each slice. The primary objective of the proposed work is to eliminate the need for further dividing slices so as not to increase the depth of the circuit. Instead, our method directly derives algebraic expressions, ensuring that parallelism is maximized (or the depth is minimized) and thereby the computation required for obtaining the desired expression is reduced.

In this paper, we propose a method that directly yields algorithmic expressions without the necessity of dividing the circuit into additional slices. This approach maximizes parallelism while effectively reducing the computational burden associated with obtaining algebraic expressions. Secondly, we optimize the resulting algebraic expressions by using circuit identities and cancellation rules to get optimized circuits. The proposed approach is outlined in Algorithm_1.

Algorithm_1: Optimization Outline

Input: - Original Quantum circuit C
Output: - Optimized Quantum circuit C^1

1) Represent Quantum Circuit C as algebraic Expression.
2) Optimize algebraic Expressions by using circuit identities and gate cancellation rules.
3) Convert optimized algebraic expression to optimized Quantum Circuit C^1

Algorithm_1: Outline of optimization procedure using algebraic expressions.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63778-0_19

3 Representing quantum circuits as Algebraic expressions

The proposed approach to represent quantum circuits as Algebraic expressions while maximizing parallelism is outlined next.

Step 1: - Identifying Number of terms*:* Given a slice with N control nodes, the number of terms in the algebraic expression will be 2^N terms. If T_i represents term i, then the slice will have the algebraic form.

$$U = \sum_{i=0}^{2^n} T_i$$

Step 2: - Representing each term: If we have a m qubit quantum circuit (q_{m-1}, q_{m-2}.....q_0), each term T_i represents the Tensor products of qubits from left to right. In each term, there will be m characters represented by C_i representing each quantum bit in the quantum circuit.

$$T_i = \otimes_{j=m-1}^{0} C_j = \quad C_{m-1}\, C_{m-2}.....C_0$$

Step 3: - Finalizing the expression: If the control bit is in state $|1\rangle$, then the target bit undergoes the corresponding target gate computation. On the other hand, if the control bit is in state $|0\rangle$, the target gate acts as an identity gate, leaving the target bit unchanged. We term that D_0 as the inactive control matrix and D_1 as the active control matrix.

If in the slice, there are k control bits represented as a_1, a_2....,a_k qubits respectively ({for all a_i} ∈ {0, 1, ... m-1}), then we have 2^k terms as mentioned in step 1 and each term in the final expression will be represented in binary notation as shown below.

$$U = \sum_{i=0}^{2^k-1} T_k \ , \ (\mathrm{k} \in \{0,1\}\ \mathrm{n})$$

For example, If there are three control nodes acting on a_1, a_2, a_3 qubits where each a_i is a control qubit, the final expression can be represented as

U = $T_0 + T_1 + T_2 + T_3 + T_4 + T_5 + T_6 + T_7 = T_{\{000\}} + T_{\{001\}} + T_{\{010\}} + \ldots\ldots + T_{\{111\}}$

1.1 In each of the terms generated from the k control bits, in the term represent each bit which equal to '0' as the inactive control qubit and '1' as the active control qubit respectively.

1.2 Step 2 presents how each term T_i can be represented as a tensor product $\otimes\ ^{0}_{j=m-1}$ $C_j = C_{m-1}\, C_{m-2}.....C_0$. In the binary representation of each term $T_{\{k\}}$, each k_p which is a control qubit represents a character in the algebraic expression. where $p \in (0 \ldots n\text{-}1)$ and n is the number of control nodes in the respective slice. Each k_p represents a character representing the a_i qubit. Therefore, the character k_p is in the C_{ai}^{th} position. In each term, if k_p is 0 then D_0 replaces the respective C_{ai}^{th} position in the term. If k_p is 1 then D_1 replaces the respective C_{ai}^{th} position in the term.

1.3 In the quantum circuit, each control node may be associated with one or more target bits. Similar to the previous step described, when considering the algebraic representation, if a target bit is associated with an inactive control node, an identity gate will be substituted at the respective target qubit position within the term. If a target bit is associated with an active control node, which is replaced by the matrix D_1, the target gate matrix will be substituted at the respective qubit position within the term. This allows for the incorporation of control gates into algebraic expression. The behavior of the control node determines whether the target gate is applied or replaced with an identity gate. By considering the status of the control node, the corresponding gate operation can be appropriately represented within the term, resulting in an accurate and comprehensive algebraic expression for the quantum

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63778-0_19

circuit. When there are multiple targets, similar steps will take place, that is, multiple targets which are associated to the same control node will be replaced at once.When dealing with multiple targets in a quantum algorithm, similar steps will occur where multiple targets associated with the same control node are replaced

Example: -

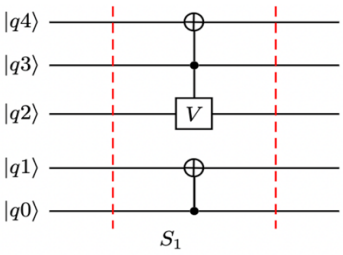

Figure 2a: Example for proposed algorithm.

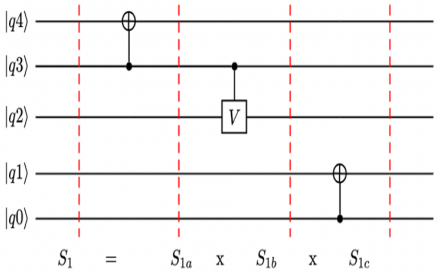

Figure 2b: Existing algorithm breaking into slices

Consider the quantum circuit with one slice S_1 and multiple quantum gates as shown in figure 2a. This example illustrates how the proposed approach works and also its efficiency in generating optimized algebraic expressions.

The algebraic expression using our proposed approach is:

1. There are two control nodes in the slice, so the number of terms in the algebraic expression is $2^2 = 4$. By step 1, $S_1 = T_0 + T_1 + T_2 + T_3$
2. Each term can be represented as a tensor product of characters (= number of qubits). Here there are five qubits. By step 2,
   $T_0 + T_1 + T_2 + T_3 = C_4C_3C_2C_1C_0 + C_4C_3C_2C_1C_0 + C_4C_3C_2C_1C_0 + C_4C_3C_2C_1C_0$
3. There are two control nodes acting on the q3 and q0 qubits, Binary indices of term indices are represented as $S_1 = T_{\{00\}} + T_{\{01\}} + T_{\{10\}} + T_{\{11\}}$
4. In the Binary representation of $T_{\{ij\}}$ where i represents qubit q_3 and j represents qubit q_0. For example, the 2nd term $T_{\{01\}}$, $ij = 01$; here $i = 0$ representing the q3 qubit and $j = 1$ representing the q0 qubit .
5. In step 1.3 replace C_3 with D_0 and C_0 with D_1 .
6. In $T_{\{01\}}$, q3 is inactive and q0 is active, the replacement process involves replacing the target nodes that are associated with q_3, namely q_2 and q_4, with the identity matrix. At the same time, substitute the target matrices associated with q_0, that is the q1 qubit, at their respective positions within the algebraic expression. The final expression is given in equation 1 and is represented in figure 2a.

$$S_1 = ID_0IID_0 + ID_0IND_1 + ND_1VID_0 + ND_1VND_1 \quad -- \quad (1)$$

Comparing equation (1) with the algebraic expression generated with the existing method [6] as shown in figure 2b.

$$S_1 = S_{1a} \times S_{1b} \times S_{1c}$$
$$= [(ID_0III + ND_1III) \times (ID_0III + ID_1VII) \times (IIIID_0 + IIIND_1)] \quad -- \quad (2)$$

We observe equation (1) is a simplified version when compared to equation 2. Hence, the proposed methodology not only offers a correct method (that is, the outcome remains the same), but also a more efficient approach by directly providing the equation to be solved.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63778-0_19

4 Optimizing the Algebraic Expressions

The next step is to optimize the algebraic expressions. Algebraic expression of quantum circuits provides a concise and simple way to represent complex quantum circuits. It uses elementary quantum operators such as X, Y, Z, H, phase shift, CX, CCX, SWAP to express the relationships between inputs and outputs. The use of these universal unitary operators ensures consistency in representing quantum circuits across different designs. This readable representation of a quantum circuit allows circuit behavior to be expressed in a natural and intuitive way, making it easier to understand and communicate quantum circuit functionality.

The quantum circuit is represented as a list of gates that are applied sequentially. The following transformation rules are then applied to optimize the quantum circuits.

To simplify or eliminate gates in a quantum circuit, remove gates that are directly next to their inverse. By rearranging the gates in the circuit, it becomes possible to transform or eliminate them. Specifically, when there is a U gate in the circuit, the optimization algorithm looks for a corresponding U† gate where U† is the adjoint. If a U† gate is found, the U gate is successfully canceled out. For some elementary gates such as X, Y, Z, H, and CNOT, U† is equal to U. In such cases, if two instances of those gates are adjacent to each other, they get cancelled out.

For two rotation gates RZ(θi) and RZ(θj) that have a shared control line or adjacent to each other, merge the two rotations to make it a single gate. This we call gate fusion.

By using techniques such as the cancellation rule and gate fusion the depth is reduced by rearranging for possible commuting quantum gates. The quantum circuit can be simplified, leading to improved performance in terms of time and size of the circuits. Until now as shown in the previous section, we have represented the quantum circuit using the most optimal algebraic expressions. In this section we extend this work in the optimization of quantum circuits by building on the algebraic expressions discussed in section 3. The proposed method uses the following steps, to begin the optimization process, we first analyze the circuit shown in Figure 3a and express it using algebraic expression notation. This algebraic expression represents the overall behavior of the circuit in terms of mathematical operations and variables. By manipulating and simplifying this expression using basic identities [1] and distributive properties of algebra, we arrive at an optimized form. Once the optimized algebraic expression is derived, the next step involves converting it back into a quantum circuit. This conversion process translates the algebraic operations and variables into corresponding quantum gates and qubits which results in figure 3b.

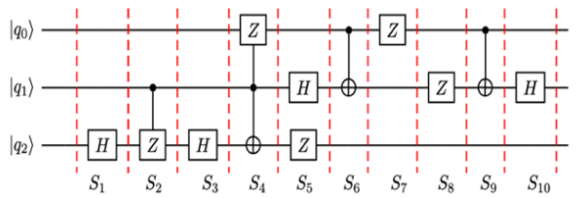

Figure 3a: Circuit C_2.

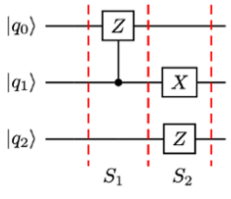

Figure 3b: Simplified version of circuit C_2

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63778-0_19

The process of optimizing the given quantum circuit C_2 involves the following steps: The unitary matrix U representing the circuit C_2 is formed by,

U= $S_1 \times S_2 \times S_3 \times S_4 \times S_5 \times S_6 \times S_7 \times S_8 \times S_9 \times S_{10}$ where each slice is given by algorithm mentioned in above section

$S_1 = IIH;$ $S_2 = ID_0I+ID_1Z;$ $S_3=IIH;$ $S_4=ID_0I+ZD_1X;$ $S_5 = IHZ;$

$S_6 = ID_0I+ID_1X;$ $S_7 = ZII;$ $S_8 =IZI;$ $S_9 = ID_0I+ID_1X;$ $S_{10} = IHI;$

$S_1 \times S_2 \times S_3 \times S_4 \times S5 \times S6 \times S7 \times S8 \times S9 \times S10 =$

$= (IIH)\ (ID_0I+ID_1Z)\ (IIH)\ (ID_0I+ZD_1X)\ (IHZ)(D_0II+D_1XI)\ (ZII)(IZI)(D_0II+D_1XI)\ (IHI)$

$=(ID_0H+ID_1\ H.Z)\ (IIH)\ (ID_0I+ZD_1X)(D_0HZ+D_1H\ .\ X\ Z)\ (ZII)(IZI)(D_0HI+D_1X.\ HI)$

$=(ID_0I+ID_1H.Z.H)\ (ID_0I+ZD_1X)(\ (D_0HZ+D_1H\ .\ X\ Z)\ (ZII)(D_0\ Z.HI+D_1\ .Z.\ H.ZI)$

$= (ID_0I+ID_1X)\ (ID_0I+ZD_1X)(D_0HZ+D_1H\ .\ X\ Z)\ (D_0\ Z.HI\ -D_1\ XI)$

$= (ID_0I + ZD_1I)\ IXZ$

the resultant unitary matrix for circuit C_2 ; $\boldsymbol{U = (ID_0\ I + ZD_1I)\ (IXZ)}$

This algebraic expression simplification process can be automated/implemented by creating an algorithm that parses algebraic expressions into Abstract Syntax Trees (AST), applying predefined simplification rules recursively until we get the same expression as the outcome and converts the simplified AST back into algebraic expression format. The expression is then converted into a quantum circuit. As a result of simplifying using our approach, the circuit is reduced by 75% in terms of the number of gates in C_2. The simplified optimized circuit is shown in Figure 3b. Instead of 10 levels there are only 2 levels resulting in improved performance and reduced error.

5 Results

Quantum circuit optimization can be measured by gate count, depth, Mitigation Overheads, Mitigation time, Average fidelity, and Execution time. Average fidelity quantifies how closely the actual gates on your device match the ideal gates. The ideal process matrix is subtracted from estimated calibration matrices (readout, Decoherence, cross talk). Error mitigation is a process for enhancing the reliability of quantum computations. The average resultant state of a given circuit is obtained by sampling. This will be done by the processor. Obviously if the circuit size is large the mitigation time and mitigation overheads increase. The circuits 3a and optimized 3b are executed on the ibm_perth processor [11]. Both circuits return the same resultant state |010> with different optimization metrics values shown in figure 4 and table 1.

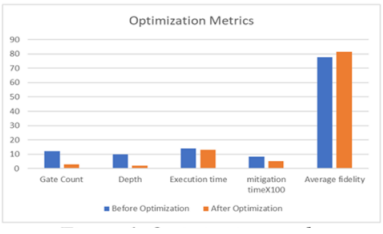

Figure 4: Optimization Results

Optimization Metrics	*Circuit before Optimization*	*Circuit after Optimization*
Gate Count	*12*	*3*
Depth	*10*	*2*
Mitigation Overheads	*88.16*	*87.35*
Mitigation time	*0.082*	*0.051*
Average fidelity	*77.87*	*81.6*
Execution time	*13.5*	*13*

Table 1: Optimization results

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63778-0_19

The results show that the proposed algebraic simplification method results in reduced Gate count, depth, execution time, mitigation time, while increasing the average fidelity. This is for a small circuit. The improvement using our proposed approach on larger circuits will be more substantial and significant.

6 Conclusion

This paper proposes a novel approach to represent quantum circuits using algebraic expressions. These expressions serve as identities to analyze and simplify quantum circuits. By applying algebraic simplification techniques, we can transform complex logical structures within circuits into more straightforward and efficient forms. This simplification process reduces circuit complexity, which in turn improves the overall performance of quantum circuits and lowers their power consumption.

References

1. Jiang, Hui-Juan, Di Long Li, Yuxin Deng and Ming Xu. “A Pattern Matching-Based Framework for Quantum Circuit Rewriting.” (2022).
2. Nam, Y., Ross, N.J., Su, Y. et al. Automated optimization of large quantum circuits with continuous parameters. npj Quantum Inf **4**, 23 (2018). https://doi.org/10.1038/s41534-018-0072-4
3. Mingyu Chen, Yu Zhang, Yongshang Li, Zhen Wang, Jun Li, and Xiangyang Li. 2022. QCIR: Pattern Matching Based Universal Quantum Circuit Rewriting Framework. In Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design (ICCAD '22)., Article 55, 1–8. https://doi.org/10.1145/3508352.3549405
4. Raban Iten, Romain Moyard, Tony Metger, David Sutter, and Stefan Woerner. 2022. Exact and Practical Pattern Matching for Quantum Circuit Optimization. ACM Transactions on Quantum Computing 3, 1, Article 4 (March 2022), https://doi.org/10.1145/3498325 .
5. Md. Mazder Rahman, Gerhard W. Dueck, and Joseph D. Horton. 2015. An Algorithm for Quantum Template Matching. J. Emerg. Technol. Comput. Syst. 11, 3, https://doi.org/10.1145/2629537
6. Steven R. Hutsell , Garrison W. Greenwood (2009) Efficient algebraic representation of quantum circuits, Journal of Discrete Mathematical Sciences and Cryptography, 12:4, 429-449, DOI: 10.1080/09720529.2009.10698246
7. França, Daniel & Garcia-Patron, Raul. (2021). Limitations of optimization algorithms on noisy quantum devices. Nature Physics. 17. 1-7. 10.1038/s41567-021-01356-3.
8. Wang, S., Fontana, E., Cerezo, M. et Noise-induced barren plateaus in variational quantum algorithms. Nat Commun 12, 6961 (2021). https://doi.org/10.1038/s41467-021-27045-6
9. Abdessaied, N., Soeken, M., Drechsler, R. (2014). Quantum Circuit Optimization by Hadamard Gate Reduction. In: Yamashita, S., Minato, Si. (eds) Reversible Computation. RC 2014. Lecture Notes in Computer Science, vol 8507. Springer, Cham. https://doi.org/10.1007/978-3-319-08494-7_12
10. Cross, Andrew W., et al. "Open quantum assembly language." arXiv preprint arXiv:1707.03429 (2017)
11. IBM quantum Composer https://quantum-computing.ibm.com/composer Dec 5th 2023.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63778-0_19