IEEE Access
The journal for rapid open access publishing

Received August 11, 2015, accepted August 29, 2015, date of publication September 15, 2015, date of current version October 8, 2015.

Digital Object Identifier 10.1109/ACCESS.2015.2478793

Non-Dominated Quantum Iterative Routing Optimization for Wireless Multihop Networks

DIMITRIOS ALANIS, (Student Member, IEEE), PANAGIOTIS BOTSINIS, (Student Member, IEEE), ZUNAIRA BABAR, SOON XIN NG, (Senior Member, IEEE), AND LAJOS HANZO, (Fellow, IEEE)
School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K.

Corresponding author: L. Hanzo (lh@ecs.soton.ac.uk)

This work was supported in part by the European Research Council Advanced Fellow Grant and in part by the Royal Society's Wolfson Merit Award.

ABSTRACT Routing in wireless multihop networks (WMHNs) relies on a delicate balance of diverse and often conflicting parameters, when aiming for maximizing the WMHN performance. Classified as a non-deterministic polynomial-time hard problem, routing in WMHNs requires sophisticated methods. As a benefit of observing numerous variables in parallel, quantum computing offers a promising range of algorithms for complexity reduction by exploiting the principle of quantum parallelism (QP), while achieving the optimum full-search-based performance. In fact, the so-called non-dominated quantum optimization (NDQO) algorithm has been proposed for addressing the multiobjective routing problem with the goal of achieving a near-optimal performance, while imposing a complexity of the order of $O(N)$ and $O(N\sqrt{N})$ in the best and worst case scenarios, respectively. However, as the number of nodes in the WMHN increases, the total number of routes increases exponentially, making its employment infeasible despite the complexity reduction offered. Therefore, we propose a novel optimal quantum-assisted algorithm, namely, the non-dominated quantum iterative optimization (NDQIO) algorithm, which exploits the synergy between the hardware and the QP for the sake of achieving a further complexity reduction, which is on the order of $O(\sqrt{N})$ and $O(N\sqrt{N})$ in the best and worst case scenarios, respectively. In addition, we provide simulation results for demonstrating that our NDQIO algorithm achieves an average complexity reduction of almost an order of magnitude compared with the near-optimal NDQO algorithm, while having the same order of power consumption.

INDEX TERMS WMHNs, quantum computing, Pareto optimality, BBHT-QSA, DHA, NDQO.

LIST OF ACRONYMS

ACO	Ant Colony Optimization
BBHT	Boyer, Brassard, Høyer and Tapp
BER	Bit Error Ratio
BF	Brute Force
BSC	Binary Symmetric Channel
BW-BBHT	Backward BBHT-QSA
CD	Classical Domain
CDMA	Code-Division Multiple Access
CF(E)	Cost Function (Evaluation)
CLT	Central Limit Theorem
CNOT	Controlled-NOT quantum gate
CPU	Central Processing Unit
DF	Decode-and-Forward
DCCP	Dynamic Coverage and Connectivity Problem
DHA	Durr-Høyer-Algorithm
DN	Destination Node
DSS	Direct-Sequence Spreading
GA	Genetic Algorithm
GPU	Graphics Processing Unit
HGR	Hybrid Geographic Routing
HIHO	Hard-Input Hard-Output
HP	Harware Parallelism
HYMN	HYbrid Multihop Network
MODE	Multi-Objective Differential Evolution
MUD	Multi-User Detection
NDQO	Non-Dominated Quantum Optimization
NDQIO	Non-Dominated Quantum Iterative Optimization
NP	Non-deterministic Polynomial-time
NSGA-II	Non-dominated Sort Genetic Algorithm II

OF	Objective Function
OFDM	Orthogonal Frequency-Division Multiplexing
OPF	Optimal Pareto Front
(G/L)OW	(Global/Local) Oracle Workspace
PF	Pareto Front
PLR	Packet Loss Ratio
QAE	Quantum Amplitude Estimation
(G/L)QCR	(Global/Local)Quantum Control Register
QC	Quantum Counting
QD	Quantum Domain
QFT	Quantum Fourier Transformation
QGOA	Quantum Genetic Optimization Algorithm
(G/L)QIR	(Global/Local) Quantum Index Register
QMA	Quantum Mean Algorithm
QoS	Quality of Service
QP	Quantum Parallelism
QPE	Quantum Phase Estimation
QR	Quantum Register
QSA	Quantum Search Algorithm
QWSA	Quantum Weighted Sum Algorithm
RN	Relay Node
SDMA	Spatial-Division Multiple Access
SISO	Soft-Input Soft-Output
SN	Source Node
SR	Self-Repair
SSH	Slow Sub-carrier Hopping
UF	Utility Function
UV	Utility Vector
VoIP	Voice over Internet Protocol
VANET	Vehicular Ad-hoc Network
WMHN	Wireless Multihop Networks
WSN	Wireless Sensor Network
XOR	Exclusive OR gate

LIST OF SYMBOLS

C	Pareto Completion Ratio
CL	Overall Route Power Dissipation
CD	Overall Route Delay
$\mathbf{f}(x)$	Utility Vector of the x-th route
$f_k(x, i)$	Lower Comparison Function Between the x-th and the i-th Routes
$\mathcal{G}$	Grover's QSA Operator
$g(x, i)$	Dominance Operator Function Between the x-th and the i-th Routes
H	Quantum Hadamard Gate
L_{dB}	Path Loss per Individual Link in dB
$L_{BBHT}^{QD,\max}$	BBHT-QSA time-out in $\mathcal{G}$ Applications
$L_{DHA}^{QD,\max}$	DHA time-out in $\mathcal{G}$ Applications
$L_{NDQIO,tot}^{T\max}$	Upper Complexity (Execution Time) Bound of the NDQIO Algorithm
$L_{NDQIO,tot}^{T,\min}$	Lower Complexity (Execution Time) Bound of the NDQIO Algorithm
$L_{NDQIO,tot}^{P\max}$	Upper Power Consumption Bound of the NDQIO Algorithm
$L_{NDQIO,tot}^{P,\min}$	Lower Power Consumption Bound of the NDQIO Algorithm
$L_{NDQO}^{tot,\max}$	Upper Complexity Bound of the NDQO Algorithm
$L_{NDQO}^{tot,\min}$	Lower Complexity Bound of the NDQO Algorithm
L_{qr}	Length of a Quantum Register in qubits
λ_c	Carrier Frequency
N	Total Number of Legitimate Routes
O	Grover's QSA Quantum Oracle Gate
P_d	Pareto Distance
P_e	Bit Error Ratio
S	Set of Legitimate Routes
T_n	n-qubit Quantum Toffoli Gate
U_f	Generic Quantum Unitary Operator implementing the function $f(x)$
U_{f_k}	Quantum Unitary Operator implementing the Comparison Operator $f_k(x, i)$
U_g	Quantum Unitary Operator implementing the Dominance Operator $g(x, i)$
$U_{g'}$	Parallel Quantum Unitary Operator implementing the Dominance Operator $g(x, i)$
U_G	Quantum Unitary Operator for Parallel Activation of Multiple $U_{g'}$ Operators
x	Index of the Legitimate Route in the Route List
$\|\psi\rangle$	Quantum State ψ

I. INTRODUCTION

Wireless Multihop Networks (WMHNs) [1] facilitate both direct and indirect communication between the source and the destination nodes, since each WMHN node is capable of relaying its message using a series of intermediate nodes to reach its destination. Explicitly, this concept can be readily applied to all networks ranging from *Wireless Sensor Networks* (WSNs) [2] and *wireless ad-hoc networks* [3] to *smart grid networks* [4]. Furthermore, each WMHN node attempts to optimize its performance in terms of different and often conflicting Quality of Service (QoS) parameters, such as the Bit Error Ratio (BER), the Packet Loss Ratio (PLR) and the end-to-end delay, while having access to a restricted amount of power. Therefore, optimal routing is essential for satisfying the aforementioned QoS criteria. Nevertheless, as the number of WMHN nodes involved escalates, the total number of potential routes increases exponentially, turning the routing optimization problem into a *Non-deterministic Polynomial-time hard* (NP-hard) one [5], hence requiring sophisticated heuristic methods. Let us now proceed by presenting the related work carried out in the field of routing.

A. RELATED WORK

A plethora of single-objective optimization techniques exist in the literature [6]–[15], each addressing different routing aspects. For instance, Zhu *et al.* [6] have proposed a routing protocol that succeeds in minimizing the energy consumption in WSNs by organizing the nodes using Hausdorff clusters [16]. Additionally, Chen *et al.* [7] have conceived a Hybrid Geographic Routing (HGR) scheme for minimizing the total energy dissipation, while satisfying the end-to-end delay constraints imposed. Abdulla *et al.* [8] have maximized the lifetime of WSNs by introducing a range of Hybrid Multihop Network (HYMN) parameters. Furthermore, Al-Rabayah and Malaney [9] have designed a hybrid routing protocol for minimizing the routing overhead imposed for example by broken links in *Vehicular Ad-hoc Networks* (VANETs). In a similar context, namely that of the *Aeronautical Ad-Hoc Networks* (AAHN) [17], Hoffmann *et al.* [10] proposed a routing scheme for AAHNs, which employed a *Genetic Algorithm* (GA) [18] for the the sake of minimizing both the transmission delay and the packet delivery ratio. Moving on to the concept of smart grids, Li *et al.* [11] proposed a multicast routing protocol for stabilizing a network of distributed energy generators. Additionally, Shah *et al.* [12] improved the link quality of the wireless sensors controlling a smart grid by employing a distributed control algorithm for jointly optimizing the network delay, the bandwidth usage and the network's reliability. Moreover, Canale *et al.* [13] conceived a joint routing and resource allocation scheme for maximizing the reliability of medium-voltage power-line networks.

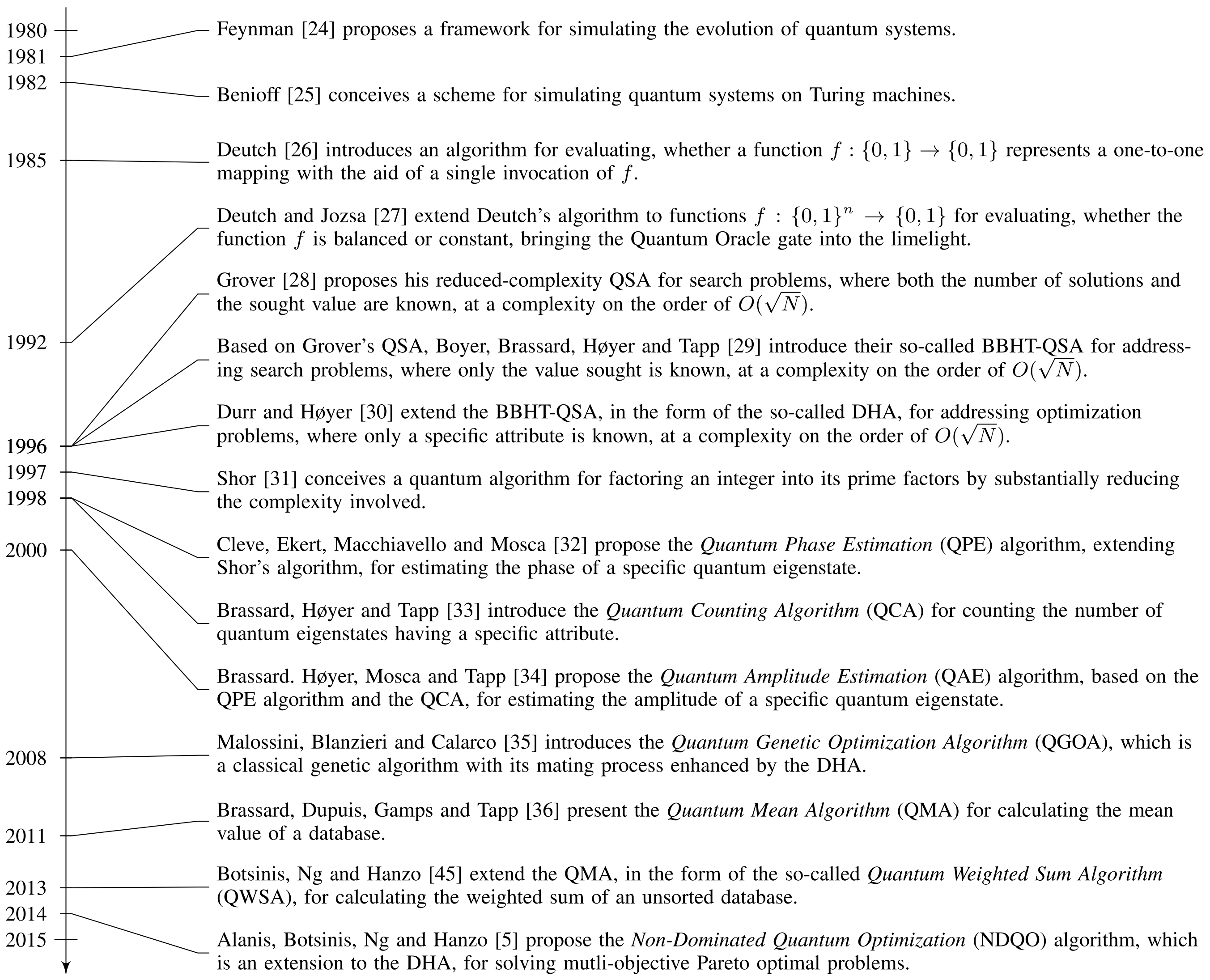
On the other hand, the potential degradation of the routing efficiency metrics can be mitigated by using a multi-objective optimization approach [19] in the context of routing problems, albeit at the expense of an increased complexity. For this reason, several studies [20]–[23], involved this multi-objective approach relying on near-optimal, evolutionary methods for addressing the associated networking aspects. To elaborate further, Yetgin *et al.* [20] have employed both the *Non-dominated Sort Genetic Algorithm II* (NSGA-II) and the *Multi-Objective Differential Evolution* (MODE) algorithm for jointly optimizing both the energy dissipation and the end-to-end delay, utilizing the concept of Pareto Optimality [19]. Camelo *et al.* [21] employed the NSGA-II for optimizing the same QoS parameters both in the context of the Voice over Internet Protocol (VoIP) and for file transfer in wireless mesh networks. Moreover, a hybrid multi-objective evolutionary algorithm has been employed by Martins *et al.* [22] for addressing the so-called Dynamic Coverage and Connectivity problem (DCCP) in WSNs exhibiting node failures.

The recent advances in quantum computing [24]–[36] and quantum information theory [37]–[39] provide us with an attractive framework of addressing NP-hard problems at a full-search-based accuracy, despite imposing a reduced complexity by exploiting the powerful concept of *Quantum Parallelism* (QP) [40]. To elaborate further, Feynman [24] proposed in 1981 a novel framework for simulating the evolution of the quantum states. In the following year, Benioff [25] proposed a technique of simulating quantum systems on Turing machines. Several years later, the effect of QP has been exploited by Deutch [26], who conceived an algorithm, named after him as the *Deutch's Algorithm*, for determining whether a binary function $f : \{0, 1\} \rightarrow \{0, 1\}$ has or has not one-to-one mapping by only using a single call of the function. An extension of this algorithm, namely the so-called *Deutch-Jozsa Algorithm* [27], was conceived for determining whether a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is balanced or constant. The *Deutch-Jozsa Algorithm* laid the foundations for the development of the so-called *Quantum Oracle* gates [41], which are quantum circuits implementing a generic mapping function $f : \{0, 1\}^N \rightarrow \{0, 1\}^M$ and they are capable of calculating all the pairs of possible inputs-outputs of f using a single call of f by exploiting the QP.

Based on these gates, Grover [28] has proposed a *Quantum Search Algorithm* (QSA), which has been shown to be optimal by Zalka [42]. This QSA is capable of finding a desired solution stored in an unsorted database by imposing a low complexity, which is on the order of $O(\sqrt{N})$, as long as both the number of valid solutions and the solution to be found are known to the optimization process. An even more powerful extension of Grover's QSA has been introduced by Boyer *et al.* [29] in the form of the so-called *Boyer-Brassard-Høyer-Tapp* QSA (BBHT-QSA), which is applicable in the specific scenario, where the actual number of valid solutions is unknown, whilst imposing the same order of complexity, namely $O(\sqrt{N})$. A further extension of the BBHT-QSA has been conceived by Durr and Høyer [30], where the *Durr-Høyer Algorithm* (DHA) is employed for identifying the extreme values of an unsorted database, while imposing a low complexity, which is on the order of $O(\sqrt{N})$. Subsequently, Malossini *et al.* [35] proposed the so-called *Quantum Genetic Optimization Algorithm* (QGOA), which constitutes a steady-state GA [43], in which the mating process is enhanced by the DHA.

Furthermore, several contributions exist, which exploit the properties of the *Quantum Fourier Transformation* (QFT) [40], [44]. In particular, Shor [31] has proposed a quantum algorithm for addressing the prime integer factorization problem at a complexity on the order of $O(\log(N)^3)$. Shor's algorithm formed the basis for the concept of *Quantum Phase Estimation* (QPE), which was proposed by Cleve *et al.* [32] and allowed the estimation of the phase of a specific quantum eigenstate. This innovation led, in turn, to the concept of both *Quantum Counting Algorithm* (QCA) [33] as well as to that of *Quantum Amplitude Estimation* (QAE) [34]. Based on these concepts, Brassard *et al.* [36] proposed the so-called *Quantum Mean Algorithm* (QMA) for calculating the mean of values found in an unsorted database at a reduced complexity. The milestones of quantum computing are summarized in the timeline of Fig. 1.

Several studies [5], [45]–[49] exist that invoke quantum algorithms for addressing diverse high-complexity telecommunications problems as an explicit benefit of the complexity reduction offered by QP. To elaborate further, Botsinis *et al.* [45] have introduced an extension of



Feynman [24] proposes a framework for simulating the evolution of quantum systems.

Benioff [25] conceives a scheme for simulating quantum systems on Turing machines.

Deutch [26] introduces an algorithm for evaluating, whether a function $f : \{0,1\} \rightarrow \{0,1\}$ represents a one-to-one mapping with the aid of a single invocation of f.

Deutch and Jozsa [27] extend Deutch's algorithm to functions $f : \{0,1\}^n \rightarrow \{0,1\}$ for evaluating, whether the function f is balanced or constant, bringing the Quantum Oracle gate into the limelight.

Grover [28] proposes his reduced-complexity QSA for search problems, where both the number of solutions and the sought value are known, at a complexity on the order of $O(\sqrt{N})$.

Based on Grover's QSA, Boyer, Brassard, Høyer and Tapp [29] introduce their so-called BBHT-QSA for addressing search problems, where only the value sought is known, at a complexity on the order of $O(\sqrt{N})$.

Durr and Høyer [30] extend the BBHT-QSA, in the form of the so-called DHA, for addressing optimization problems, where only a specific attribute is known, at a complexity on the order of $O(\sqrt{N})$.

Shor [31] conceives a quantum algorithm for factoring an integer into its prime factors by substantially reducing the complexity involved.

Cleve, Ekert, Macchiavello and Mosca [32] propose the *Quantum Phase Estimation* (QPE) algorithm, extending Shor's algorithm, for estimating the phase of a specific quantum eigenstate.

Brassard, Høyer and Tapp [33] introduce the *Quantum Counting Algorithm* (QCA) for counting the number of quantum eigenstates having a specific attribute.

Brassard. Høyer, Mosca and Tapp [34] propose the *Quantum Amplitude Estimation* (QAE) algorithm, based on the QPE algorithm and the QCA, for estimating the amplitude of a specific quantum eigenstate.

Malossini, Blanzieri and Calarco [35] introduces the *Quantum Genetic Optimization Algorithm* (QGOA), which is a classical genetic algorithm with its mating process enhanced by the DHA.

Brassard, Dupuis, Gamps and Tapp [36] present the *Quantum Mean Algorithm* (QMA) for calculating the mean value of a database.

Botsinis, Ng and Hanzo [45] extend the QMA, in the form of the so-called *Quantum Weighted Sum Algorithm* (QWSA), for calculating the weighted sum of an unsorted database.

Alanis, Botsinis, Ng and Hanzo [5] propose the *Non-Dominated Quantum Optimization* (NDQO) algorithm, which is an extension to the DHA, for solving mutli-objective Pareto optimal problems.

FIGURE 1. **Timeline of quantum computing milestones.**

the QMA algorithm, namely the *Quantum Weighted Sum Algorithm* (QWSA), which was employed for *Soft-Input Soft-Output Multi-User Detection* (SISO-MUD) in the context of *Code Division Multiple Access* (CDMA) systems. Additionally, the impact of premature termination of the DHA iterations used for *Hard-Input Hard-Output Multi-User Detection* (HIHO-MUD) has been investigated by the same authors [46] in the context of *Spatial Division Multiple Access* (SDMA) systems, while an improved version of this algorithm invoked for SISO-MUD has been proposed by Botsinis *et al.* in [47] for *Direct Sequence Spreading* (DSS) and *Spatial-Division Multiple Access - Orthogonal Frequency-Division Multiplexing* (SDMA-OFDM) systems. As for the multi-objective routing problem, to the best of the authors' knowledge, there exists only a single comprehensive study by Alanis *et al.* [5] benefiting of the QP, where the so-called *Non-Dominated Quantum Optimization* (NDQO) algorithm has been introduced for jointly optimizing the transmission route in terms of the achievable BER, the total power dissipation and the end-to-end delay using the principle of Pareto Optimality of evaluating the routes, while imposing a complexity on the order of $O(N)$ and $O(N\sqrt{N})$ in the best- and the worst-case scenario, respectively.

Apart from those benefiting from the QP, there are several contributions [50]–[56], which use another realm of parallelism, namely that of the *Hardware Parallelism* (HP), for achieving a complexity reduction, while addressing the routing problem. The complexity reduction offered by HP has been mainly enabled through the use of *Graphics Processing Units* (GPU) [57] architectures for general purpose programming [58] apart from *Central Processing Units* (CPU). As for the routing problem, Han *et al.* [50] introduced a hybrid GPU-CPU concurrent framework for routers, which offered a substantial complexity reduction in the context of the global routing problem. Additionally, both Mu *et al.* [51] and Zhao *et al.* [52] proposed their routing

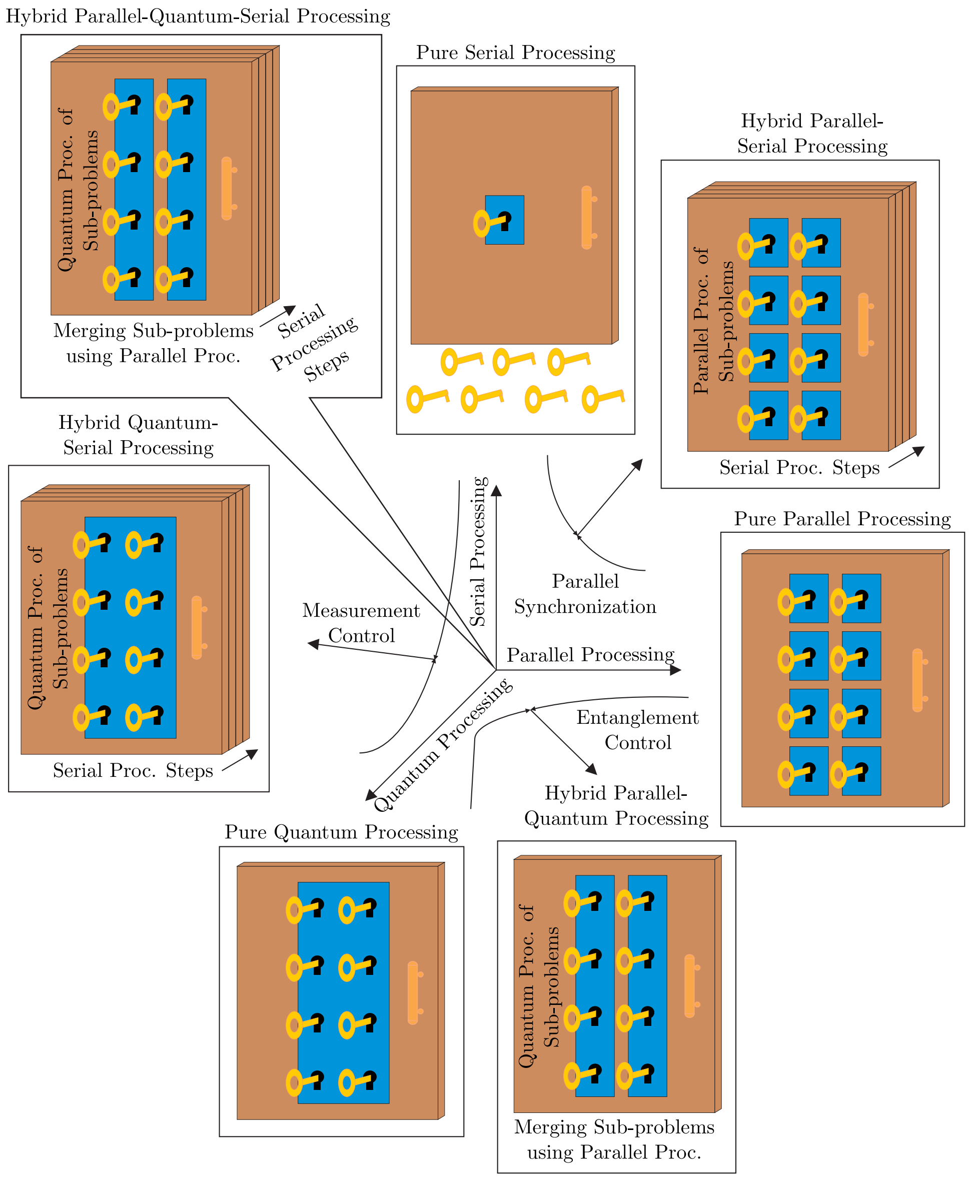

Hybrid Parallel-Quantum-Serial Processing
Quantum Proc. of Sub-problems
Merging Sub-problems using Parallel Proc.
Serial Processing Steps
Pure Serial Processing
Hybrid Parallel-Serial Processing
Parallel Proc. of Sub-problems
Serial Proc. Steps
Hybrid Quantum-Serial Processing
Quantum Proc. of Sub-problems
Serial Proc. Steps
Serial Processing
Parallel Synchronization
Measurement Control
Parallel Processing
Pure Parallel Processing
Quantum Processing
Entanglement Control
Hybrid Parallel-Quantum Processing
Pure Quantum Processing
Quantum Proc. of Sub-problems
Merging Sub-problems using Parallel Proc.

FIGURE 2. **Types of processing; inspired by [45, Fig. 1].**

algorithms, which were tailored for *Internet Protocol* (IP) routers having GPU architectures. In a similar context, namely that of the *Travelling Salesman Problem* (TSP), Uchida *et al.* [54] conceived a parallel implementation of the *Ant Colony Optimization* (ACO) algorithm, while Cekmez *et al.* [55] deployed a parallel version of the GA, both for addressing the TSP problem using GPUs.

B. CONTRIBUTIONS AND PAPER STRUCTURE

Based on the aforementioned contributions conceived for routing problems, three popular types of processing have been advocated - namely *serial processing*, *parallel processing* relying on hardware parallelism and *quantum processing* relying on quantum parallelism - as portrayed in Fig. 2, which might be visualized for the sake of simplicity using the paradigm[1] of unlocking a specific lock. In *serial processing* the full set of available keys has to be checked sequentially for each keyhole to ascertain as to whether they do or do not unlock the door. This type of processing is referred to as "Pure Serial Processing" in Fig. 2. The employment of graphics processing units in routing [51]–[55] has unveiled a new perspective, where all the keys can be simultaneously inserted into identical keyholes for unlocking the door; this type of process is referred to as "Pure Parallel Processing" in Fig. 2. Nevertheless, there are practical cases, where the parallel processes have to be synchronized [50], [54], [55], leading to an

[1]We note that in Fig. 2 the keys symbolize the set of routes, while the event of inserting a specific key in the keyhole represents processing the corresponding route.

inevitable decomposition of the task into a series of carefully coordinated parallel steps. This type of processing is referred to as ''Hybrid Parallel-Serial Processing'' in Fig. 2, where the consecutive doors denote the decomposition of the task. Subsequently, quantum processing has been brought to the limelight, as a benefit of the advances in quantum computing [24]–[36]. In fact, there are algorithms relying solely on the philosophy of QP [26]–[28], [31], [33], which are classified as ''Pure Quantum Processing'' in Fig. 2. Explicitly, a ''quantum'' keyhole is represented as an elaborate single lock having multiple keyholes for portraying the principles of the QP [41]. On the other hand, some other quantum algorithms [5], [29], [30] decompose the overall task into sequential sub-problems, which are individually handled with the aid of the QP. This decomposition is inherently necessary, since these algorithms involve the *measurement* or *observation* operation [40], which terminates the quantum processes by collapsing the effect of the QP [41]. This type of processing is termed as ''Hybrid Quantum-Serial Processing'' in Fig. 2. Additionally, some independent processes may be simultaneously invoked for the sake of achieving either a further complexity reduction by benefiting both from the QP and from the HP, or a more coherent *entanglement* [41] of the outputs of the independent processes. This specific case is referred to as ''Hybrid Parallel-Quantum Processing'' in Fig. 2. Nevertheless, further decomposition of the overall task into sequential steps may be inevitable due to the *measurement* or *observation* operations that the task may involve. Therefore, this latter type of processing involves a synergy of all the potential processing types and it is hence referred to as ''Hybrid Parallel-Quantum-Serial Processing'' in Fig. 2.

Returning to our multi-objective routing problem, namely to the NDQO algorithm [5] - albeit near-optimal, when compared to the full-search-based method - offers fruitful ground for improvement, since it is classified as a ''Hybrid Quantum-Serial Processing'' algorithm. This suggests that a further reduction of its complexity is possible by exploiting the potential synergies between the QP and the HP. Furthermore, one of its features is that its complexity is on the order of $O(N)$ [5]. Whilst this complexity increase is indeed much more moderate than the exponentially escalating *Maximum Likelihood* (ML) complexity, this linear complexity increase may become prohibitive in the context of the NDQO algorithm for high dimensionality problems, where the total number of routes is excessively high [20]. Therefore, with the goal of achieving a further complexity reduction, we propose a novel quantum-assisted algorithm, namely the *Non-Dominated Quantum Iterative Optimization* (NDQIO) algorithm. Our contributions related to the latter algorithm may be summarized as follows:

1) *We have developed a novel framework both for combining quantum unitary operators and for activating them in parallel, with the goal of achieving a further reduction in the complexity by exploiting the synergies between QP and HP.*
2) *The proposed NDQIO algorithm exploits this parallelism with the aid of the algorithms of [29] and [30] for finding the optimum of a multi-objective routing problem in WMHNs. We have also derived the algorithm's upper and lower complexity bounds.*
3) *We have further reduced the complexity of the NDQO algorithm by introducing the novel element of elitism, which allows the NDQIO algorithm to be terminated once it concludes that the entire OPF has been identified.*
4) *We have characterized the performance versus complexity of the NDQIO algorithm and have demonstrated that it achieves the full-search-based optimal performance at a normalized complexity, which is several orders of magnitude lower than that of the NDQO algorithm.*

The rest of this paper is organized as follows. In Section II, we present the WMHN architecture along with the optimization objectives considered, while in Section III we provide an introduction to quantum-assisted optimization algorithms, leading to a succinct characterization of the NDQO algorithm. In Section IV, we present our parallel quantum oracle design conceived for the NDQIO algorithm, followed by its performance versus complexity trade-offs as well as by our conclusions in Sections V and VI, respectively. We note that a more detailed structure of this paper is portrayed in Fig. 3.

Notation: Throughout this paper, the lower (upper) boldface letters represent vectors (matrices), while the superscripts $()^{\dagger}$ and $()^{T}$ denote the complex conjugate and simple matrix or vector transposition, respectively. Moreover, the upper case italic letters denote the transfer matrices of quantum unitary operators. A single subscript in the quantum register state is used for the global quantum registers of the quantum circuit, whereas two subscripts, separated by comma, are used for the local quantum registers. Additionally, in the discussion of the algorithms the notation ''Step $X.Y$'' is used, in order to refer to the Y-th step of the X-th algorithm.

II. SYSTEM OVERVIEW

We have adopted the network model presented in [5], where we have considered the route's overall *Bit Error Ratio* (BER), its overall power dissipation as well as its overall delay as our *Utility Functions* (UF). To elaborate further, the WMHN examined is a fully interconnected network and its coverage area is a (100×100) m^2 square block, with the source node (SN) and the destination node (DN) located at the block's opposite corners. The relay node (RN) locations within the block are random, obeying a uniform distribution. We have imposed the plausible constraint that the routes must not form loops, i.e. each RN is only visited once, hence avoiding potentially excessive power consumption or PLR [20]. An 8-node WMHN topology is exemplified in Fig. 4. We have assumed that the WMHN is coordinated by a cluster head node, which is the DN relying on a quantum computer.

- List of Acronyms
- List of Symbols
- I Introduction
  - I-A Related Work
  - I-B Contributions and Paper Structure
- II System Overview
- III Fundamentals of Quantum Computing
  - III-A Quantum Computing Postulates
  - III-B Quantum Search Algorithms
    - III-B1 Grover's QSA
    - III-B2 BBHT-QSA
    - III-B3 DHA
    - III-B4 NDQO Algorithm
- IV Design Methodology
  - IV-A Parallel Oracle Design
  - IV-B NDQIO Algorithm
    - IV-B1 Initialization Process
    - IV-B2 Seeking the Next Optimal Route
    - IV-B3 OPF Repair Process
- V A Detailed 5-Node Example
- VI Computational Accuracy versus Complexity Discusions
  - VI-A NDQIO Complexity
  - VI-B NDQIO Computational Accuracy Performance
- VII Conclusions

FIGURE 3. **The structure of this paper.**

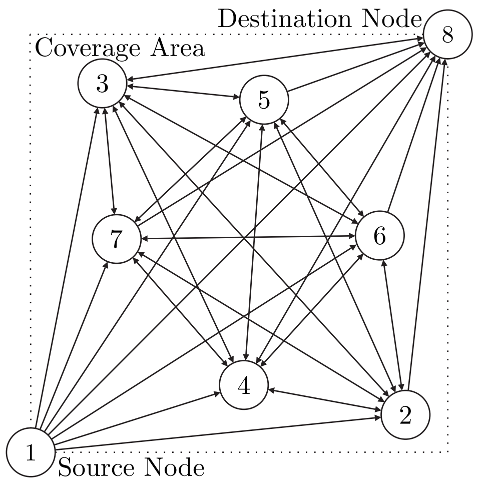


FIGURE 4. **The 8-node WMHN topology considered [5]. The WMHN is controlled by a cluster head (DN), which is assumed to be in possession of a quantum computer.**

Moreover, we have assumed that the nodes transmit their messages using QPSK modulation over *uncorrelated Rayleigh channels* [59] using the classic *Decode-and-Forward* (DF) scheme [60]. Hence, for the calculation of overall BER, we have utilized the two stage *Binary Symmetric Channel* (BSC) method presented in [5], where each link's BER P_e versus the *Bit-Energy-to-Noise Ratio* E_b/N_0 is given by the formula [59]:

$$P_e = \frac{1}{2}\left(1 - \sqrt{\frac{E_b/N_0}{E_b/N_0 + 1}}\right), \tag{1}$$

while the overall BER is given by the recursive formula of [5]:

$$P_{e,12} = P_{e,1} + P_{e,2} - 2P_{e,1}P_{e,2}. \tag{2}$$

As for the interference experienced at each receiver node, its power N_0 is assumed to obey a random Gaussian distribution, owing to the *Central Limit Theorem* (CLT) [59], with its mean set to -90 dBm and its standard deviation to 10 dB. Still referring to the physical layer, each node transmits at a power set to $P_{Tx} = 20$ dBm. The transmitted signal experiences path-loss obeying the *inverse-power path-loss model* [59], having a *path-loss exponent* of $\alpha = 3$, which is formulated as:

$$L_{dB} = P_{Tx} - P_{Rx} = 10\alpha \log\left(\frac{4\pi d}{\lambda_c}\right) \text{ [dB]}, \tag{3}$$

where L_{dB} is the path-loss of an individual link, d corresponds to the distance between the nodes and λ_c is the carrier's wavelength. The carrier frequency was set to $f_c = 1.8$ GHz, corresponding to a wavelength of $\lambda_c \cong 0.167$ m. Then, assuming that the power dissipation of signal processing is negligible compared to the power dissipation owing to the path-loss, the route's total power dissipation becomes proportional to the linear-domain sum of its constituent path-losses.

TABLE 1. WMHN specification parameters [5].

Network Coverage Area	(100×100) m^2 Square Block
Modulation	QPSK
Mean Node Interference, μ_I	-90 dBm
Node Interference Std, σ_I	10 dB
Power Consumption Model	Inverse-Power Path-Loss Model with $\alpha = 3$
Transmission Power	20 dBm
Carrier Frequency	1.8 GHz
Delay Unit	Single Hop

Moving on to the network layer, for the sake of simplicity, we have assumed that all the messages are forwarded by the RNs instantly as soon as they are decoded. Therefore, the route's overall delay is quantified in terms of the number of hops incorporated by a specific route. The network specifications are summarized on Table 1. Finally, the optimization *Utility Vector* (UV) $\mathbf{f}(x)$ is defined as [5]:

$$\mathbf{f}(x) = \left[P_{e,x}, CL_x, CD_x\right], \tag{4}$$

where $P_{e,x}$, CL_x and CD_x correspond to the overall BER, the overall power dissipation and the overall delay of the x-th route, respectively. Therefore, we can characterize the routes based on their respective UVs using the principle of *Pareto Optimality*, which is incorporated by Defs. 1, 2 and 3.

Definition 1 (Pareto Dominance [20]): A particular route-solution x_1 associated with the UV $\mathbf{f}(x_1) = [f_1(x_1), \ldots, f_n(x_1)]$ dominates another route-solution x_2 having the UV $\mathbf{f}(x_2) = [f_1(x_2), \ldots, f_n(x_2)]$ if and only if $\mathbf{f}(x_1) \succeq \mathbf{f}(x_2)$, i.e. we have $f_i(x_1) < f_i(x_2)\ \forall\, i \in \{1, \ldots, n\}$, where n corresponds to the number of optimization objectives. The operator $\succeq$ is often referred to as *dominance operator*.

Definition 2 (Pareto Optimality [20]): A particular route-solution x_1 associated with the UV $\mathbf{f}(x_1) = [f_1(x_1), \ldots, f_n(x_1)]$ is said to be Pareto optimal if and only if $\nexists\ x\ :\ \mathbf{f}(x) \succeq \mathbf{f}(x_1)$, i.e. there exists no solution that dominates x_1. The Pareto Optimal route-solutions form a front that is often referred to as the *Optimal Pareto Front* (OPF).

Definition 3 (Pareto Distance [5]): Given the set S of all the eligible route-solutions and a particular route-solution x_i belonging to the set $x_i \in S$, its distance from the OPF is defined in terms of the probability P_d of being dominated by the other solutions of S. This is formally formulated as [5]:

$$P_d(x_i) = \frac{\#\{\mathbf{f}(x_j) \succeq \mathbf{f}(x_i),\ \forall j,\ i \in \{1, \ldots, |S|\}\}}{|S|}, \tag{5}$$

where the operator $\#\{\cdot\}$ quantifies the number of times that the condition in the curly brackets is satisfied, while the operator $|\cdot|$ represents the total number of elements of a set and $\mathbf{f}(\cdot)$ is the UF vector defined in (4).

III. FUNDAMENTALS OF QUANTUM COMPUTING

A. QUANTUM COMPUTING POSTULATES

Before proceeding with the portrayal of our proposed algorithm, we will briefly present the main concepts of quantum computing. The state $|\phi\rangle$ of a quantum system is given by [41]:

$$|\phi\rangle = \sum_{i=0}^{M-1} \varphi_i\, |\phi_i\rangle = (\phi_0, \phi_0, \ldots, \phi_{M-1})^T, \tag{6}$$

where the complex valued φ_i represents the amplitude of the *basis state* $|\phi_i\rangle$ and there is a total of 2^M basis states. The squared modulus $|\varphi_i|^2$ of the amplitude φ_i corresponds to the probability of observing the quantum system in the basis state $|\phi_i\rangle$, and hence it should satisfy the normalization constraint of:

$$\sum_{i=0}^{M-1} |\varphi_i|^2 = 1. \tag{7}$$

Based on Eq. (6), it is possible to form a *Quantum Register* (QR) based on individual *quantum bits* (qubits). For example, assuming a QR consisting of $L_{qr} = 2$ qubits, the quantum system state $|\psi\rangle$ could be expressed as the tensor product of the states $|\psi_i\rangle$ of the individual qubits as [40]:

$$\begin{aligned} |\psi\rangle &= |\psi_1\rangle_1\, |\psi_2\rangle_2 && (8)\\ &= (\alpha\,|0\rangle_1 + \beta\,|1\rangle_1)\,(\gamma\,|0\rangle_2 + \delta\,|1\rangle_2) && (9)\\ &= \underbrace{\alpha\gamma}_{a_{00}}\,|00\rangle + \underbrace{\alpha\delta}_{a_{01}}\,|01\rangle + \underbrace{\beta\gamma}_{a_{10}}\,|10\rangle + \underbrace{\beta\delta}_{a_{11}}\,|11\rangle, && (10) \end{aligned}$$

where the amplitudes a_{ij} should satisfy the normalization constraint of Eq. (7).

The main challenge of quantum computing is that the amplitudes φ_i of Eq. (6) are not accessible for an external observer of the quantum system. This challenge stems from *Heisenberg's Uncertainty Principle* [41]. Explicitly, an observation of the quantum system would result in collapsing the superimposed state into the observable one, which would inherently be one of its basis states [45].

Furthermore, the evolution of the state $|\psi\rangle$ of a quantum system versus time may be characterized by a set of unitary transformations, which is formally expressed as:

$$|\psi\rangle = U\,|\phi\rangle, \tag{11}$$

where U is a unitary matrix, i.e. we have $U^{-1} = U^\dagger$ and $U^\dagger$ corresponds to the complex conjugate matrix of U. In fact, there exists a vast range of these operators, which are often referred to as *quantum gates*. One of the most common gates is the *Hadamard gate* H, which has a single-qubit transfer matrix of:

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \tag{12}$$

that is mainly used for mapping the ground state $|0\rangle$ to the superposition of the states $|0\rangle$ and $|1\rangle$. Hence, its effect is formally formulated as [40]:

$$|0\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \equiv |+\rangle, \tag{13}$$

$$|1\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \equiv |-\rangle, \tag{14}$$

where the states $|+\rangle$ and $|-\rangle$ are the so-called *Bell states* [41].

Apart from the above simple base-line operations, it is possible to define gates that perform controlled operations. A commonly used gate of this type is the *Controlled-NOT* (CNOT) gate [40]. It performs the *Exclusive OR* (XOR) operation of its two inputs storing the resultant output on the second qubit or QR. Thus, it may be viewed as the quantum counterpart of the classic XOR gate and its function is formulated as:

$$|c\rangle\,|t\rangle \xrightarrow{\text{CNOT}} |c\rangle\,|c \oplus t\rangle, \tag{15}$$

where the state $|c\rangle$ is often referred to as the *control* register, while $|t\rangle$ is the *target* register. Explicitly, the CNOT gate belongs to a broader family of quantum gates, which are commonly known as unitary operators U_f [41] and implement a specific binary function $f : \{0, 1, \ldots, N-1\} \to \{0, 1\}$ in the quantum domain. A generic quantum circuit of these gates is shown in Fig. 5; due to the superposition of states of the QR $|x\rangle_1$ it is possible to carry out the function's calculations in parallel [61], which is the main advantage of quantum computing. Their operation is formulated as:

$$|x\rangle_1\,|0\rangle_2 \xrightarrow{U_f} |x\rangle_1\,|0 \oplus f(x)\rangle_2 \equiv |x\rangle_1\,|f(x)\rangle_2\,. \tag{16}$$

We note that the subscripts of the "ket" operators are used for distinguishing the two inputs of the QRs. These unitary operators are the main constituent component of the so-called *Quantum Oracles*[2] [41]. Therefore, the QR $|x\rangle_1$ is often referred to as a *Quantum Index Register* (QIR), since it points to the indices of the input states, while the second input is commonly known as the *Oracle Workspace* (OW), since all the Oracle operations are carried out in this QR.

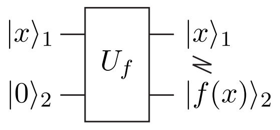


FIGURE 5. **Quantum circuit implementing a specific function $f(x)$; the subscripts of the "kets" are used in order to distinguish the two input QRs and the hashed line denotes the *entanglement* between the two output QRs.**

[2]Quantum Oracles are quantum gates that implement a binary function $f(x) : \{0, \ldots, N-1\} \to \{0, 1\}$ and are employed in QSAs for identifying the valid solutions, i.e. the solutions where we have $f(x) = \delta$, with δ being the value sought by the QSA. They then "mark" these valid solutions by flipping the sign of the respective state [5]. Inherently, the unitary operator U_f is invoked inside the oracle gate to implement the aforementioned binary function $f(x)$.

A notable property of the unitary operators is the *quantum entanglement* [41] of the output states. To elaborate further, if we assume that the QIR is in the superposition of all states, i.e. we have $|x\rangle_1 = \sum_i |i\rangle/\sqrt{N}$, where N is the total number of input states considered, the output of the operator U_f, considering both the QIR and the OW, will be in the superposition of composite states, i.e. we have:

$$|x\rangle_1\,|f(x)\rangle_2 = \frac{1}{\sqrt{N}} \sum_i |i\rangle_1\,|f(i)\rangle_2. \tag{17}$$

Eq. (17) suggests that if a *partial measurement* [45] is carried out concerning the QIR, then the state of the OW will also collapse to the state $|f(i)\rangle_2$, assuming the observable state $|i\rangle_1$ in the QIR. A direct consequence of the *quantum entanglement* is the so-called *no-cloning theorem* [41], which dictates that it is physically infeasible to clone the state of a quantum system.

B. QUANTUM SEARCH ALGORITHMS

Having defined the framework, in which quantum systems function, let us now proceed by presenting three famous QSAs, which form the basis of our NDQIO algorithm, along with the NDQO algorithm, which we will use for benchmarking the NDQIO algorithm.

1) GROVER'S QSA

Grover's QSA [28] is based on the assumption of considering a search problem, where both the number of solutions t and the actual solution value of $f(x) = \delta$ are known to the optimization process. Assuming $N = 2^n$ potential solutions in total, its function relies on the so-called *Grover operator* $\mathcal{G} = HP_0HO$, where H corresponds to a n-qubit *Hadamard Gate*, P_0 is a quantum gate that simply flips the phase of all the states except for the phase of $|0\rangle^{\otimes n}$ state, i.e. we have $|x\rangle \xrightarrow{P_0} -|x\rangle$ if $|x\rangle \neq |0\rangle^{\otimes n}$, while O is a quantum oracle gate, which implements a function $f(x)$ and "marks" the solutions in the database, in other words the states $|x\rangle$ that satisfy the condition $f(x) = \delta$, by flipping their phase, i.e we have $|x\rangle \xrightarrow{O} -|x\rangle$ if and only if $f(x) = \delta$. The rest of the states are left intact. The $\mathcal{G}$ operator attains the property of increasing the modulus of the valid solutions amplitudes, while reducing the modulus of the non-valid ones [28]. The input state $|y\rangle$ is initialized to the equally initialized superimposition of all the legitimate states, i.e. we have:

$$|y\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle, \tag{18}$$

and the $\mathcal{G}$ operator is applied L consecutive times. Then, the output state $|y'\rangle = \mathcal{G}^L\,|y\rangle$ is measured and the observable state $|x_s\rangle$ corresponding to the search problem solution is exported. The optimal number of $\mathcal{G}$ applications is set to $L_{opt} = \left\lfloor \frac{\pi}{4}\sqrt{\frac{N}{t}} \right\rfloor$ [29], yielding a probability P_s of successfully spotting a solution in the database, which is equal to $P_s = \sin^2\left[(2L_{opt}+1)\theta\right]$, where $\theta = \arcsin\sqrt{(t/N)}$.

Since the O operator invokes the function $f(x)$ once, Grover's QSA imposes a complexity on the order of $O(\sqrt{(N/t)})$ [28].

Algorithm 1 Improved Boyer-Brassard-Høyer-Tapp QSA [5]

```
1:  Import reference route index $i$.
2:  Set $m \leftarrow 1$, $\lambda \leftarrow 6/5$ and $L_{BBHT}^{QD} \leftarrow 0$, $L_{BBHT}^{CD} \leftarrow 0$.
3:  Choose $L$ uniformly from the set $\{0, \ldots, \lfloor m \rfloor\}$.
4:  Apply the $\mathcal{G}$ operator $L$ times starting from the initial state
    $|\psi\rangle$ in (18), resulting in the final state $|x_f\rangle = \mathcal{G}^L |\psi\rangle$.
5:  Observe $|x_f\rangle$ in the QD and obtain $|j\rangle$.
6:  Compute $g(i, j)$ in the CD.
7:  Update $L_{BBHT}^{CD} \leftarrow L_{BBHT}^{CD} + 1$ and $L_{BBHT}^{QD} \leftarrow L_{BBHT}^{QD} + L$.
8:  if $g(i, j) = \delta = 1$ or $L_{BBHT}^{QD} \geq L_{BBHT}^{QD,\max}$ then
9:      Set $x_s \leftarrow j$, output $x_s$, $L_{BBHT}^{CD}$, $L_{BBHT}^{QD}$ and exit.
10: else
11:     Set $m \leftarrow \min\left\{\lambda m, \sqrt{N}\right\}$.
12:     if $m = \sqrt{N}$ then
13:         Choose $L$ uniformly from the set $\{1, \ldots, \lfloor m \rfloor\}$ and
            go to step 4.
14:     else
15:         Go to step 3.
16:     end if
17: end if
```

2) BOYER-BRASSARD-HØYER-TAPP QSA

The *Boyer-Brassard-Høyer-Tapp* QSA (BBHT-QSA) [29] is the extension of Grover's QSA addressing search problems, in which the actual number of solutions t is unknown to the optimization process. Therefore, this QSA, which is formally stated in Alg. 1, can be readily applied in our search problem, where the number of route solutions is unknown. For this reason, we will define the function $g(x, i)$ implemented by the unitary operator U_g as follows [5]:

$$g(x, i) \equiv \bigcap_{k=1}^{3} f_k(x, i) = \begin{cases} 1, & \mathbf{f}(x) \succeq \mathbf{f}(i) \\ 0, & \mathbf{f}(x) \not\succeq \mathbf{f}(i), \end{cases} \tag{19}$$

where $f_k(x, i)$ corresponds to the lower-comparison-check operator between the x-th and the i-th route-solutions in terms of the k-th objective of optimization problem, which is defined as:

$$f_k(x, i) = \begin{cases} 1, & f_k(x) < f_k(i) \\ 0, & f_k(x) \geq f_k(i). \end{cases} \tag{20}$$

It is clear from Eq. (19) that the function $g(x, i)$ implements the dominance operator. Consequently, we may invoke the BBHT-QSA for identifying a route-solution that dominates the reference route having the index i. Since the number of valid route-solutions is unknown, a different method is employed for the selection of the number L of $\mathcal{G}$ applications. To elaborate further, the number L of $\mathcal{G}$ applications is selected randomly from a uniform distribution spanning across a specific range, which is expanded, when the observed state $|j\rangle$ is not a valid route-solution, i.e. if we have $g(j, i) = 0$. This range is initialized to the set $\{0, \lfloor m \rfloor\}$, where we have $m = 1$, and its specific range expansion relies on increasing the upper bound m by a factor of $\lambda = 6/5$ [29] up to a maximum of $m = \sqrt{N}$ [29], corresponding to the case where we only have $t = 1$ valid route-solution [28]. Therefore, given the reference route-solution i, the BBHT-QSA selects the number L of $\mathcal{G}$ applications from the initial range and employs the $\mathcal{G}^L$ operator to the initial state $|y\rangle$, as defined in Eq. (18). If a route-solution, dominating the reference solution, is found, the BBHT-QSA is terminated and exports the route-solution found. Otherwise, the range is consistently expanded by the factor of $\lambda = 6/5$ and the process is repeated until a valid route-solution is identified. A further limitation is imposed on the maximum total number $L_{BBHT}^{QD,\max}$ of $\mathcal{G}$ applications as a second termination condition, which is set to $L_{BBHT}^{QD,\max} = 4.5\sqrt{N}$, providing $\sim$100% probability of successfully identifying a valid route-solution [29], as long as there exists one. This yields a complexity on the order of $O(\sqrt{N})$ in terms of U_g activations. Finally, in the special case, where no valid route-solutions exist, i.e. the reference route-solution is itself the optimal one, the BBHT-QSA complexity, considering the evaluation of the function $g(x, i)$ both in the *Quantum Domain* (QD) and in the *Classic Domain* (CD), is bounded by [5]:

$$L_{BBHT}^{tot,\min} = 4.5\sqrt{N} + \log_{\lambda}\left(4.5\,\frac{\lambda - 1}{m}\,\sqrt{N} + 1\right) + 1, \tag{21}$$

$$L_{BBHT}^{tot,\max} = 10\sqrt{N} + \log_{\lambda}\sqrt{N} - 1, \tag{22}$$

where again we have $\lambda = 6/5$ [29].

3) DURR-HØYER ALGORITHM

The *Durr-Høyer Algorithm* (DHA) [30] constitutes a further extension to the BBHT-QSA [29] and it is applicable to either the minimization or the maximization of single-objective problems. In this case, neither the number t of valid solutions nor the valid solution value δ itself is known to the optimization process. This algorithm is directly applicable to our scenario, since it can be employed for identifying the specific route-solutions that are globally optimal in terms of a single-objective UF and, thus, they are Pareto optimal. We note that the DHA is formally stated in Alg. 2.

Initially, the DHA randomly selects a reference route-solution from the entire set of all the legitimate ones and activates a BBHT-QSA process for the sake of finding a route-solution that exhibits a lower UF value than that of the reference route. If the route-solution exported by the BBHT-QSA process is a valid route-solution, the reference route is confirmed to the output of the BBHT-QSA process and a new BBHT-QSA process is activated. This procedure is repeated until the BBHT-QSA process fails to export a valid route-solution or the maximum affordable number L_{DHA}^{QD} of QD queries in the QD queries is exhausted, which is set to $L_{DHA}^{QD} = 22.5\sqrt{N}$ [30], yielding a $\sim$100% probability of successfully identifying the globally optimal route-solution.

Algorithm 2 Improved Durr-Høyer Algorithm [47]

```
1: Choose a reference index $0 \leq y' \leq N-1$ randomly from the uniform distribution.
2: Set $L_{DHA}^{QD} \leftarrow 0$.
3: repeat
4:     Set $y \leftarrow y'$.
5:     Define the quantum oracle implementing the binary function $f_k(x, i)$ of Eq. (20) and set $i \leftarrow y$.
6:     Invoke the BBHT-QSA process of Alg. 1 with input the function $f_k(x, y)$ and output the index $y'$, using $L_{BBHT}^{QD}$ CFEs.
7:     Set $L_{DHA}^{QD} \leftarrow L_{DHA}^{QD} + L_{BBHT}^{QD}$.
8: until $f_k(y', y) \neq 1$ or $L_{DHA}^{QD} \geq \lceil 22.5\sqrt{N} \rceil$
9: Output $y$ and exit.
```

Algorithm 3 Non-Dominated Quantum Optimization Algorithm [5]

```
1: Initialize solution flag vector, $\mathcal{F}$, to zero.
2: Initialize $OPF = \varnothing$.
3: for $i = 0$ to $N-1$ do
4:     if $\mathcal{F}_i = 0$ then
5:         if $|OPF| > L_{BBHT}^{\max}$ or $\nexists j \in OPF : \mathbf{f}(j) \preceq \mathbf{f}(i)$ then
6:             Set $l \leftarrow i$.
7:             repeat
8:                 Set $k \leftarrow l$.
9:                 Define the oracle function $g(k, x)$ from (19).
10:                Invoke the BBHT-QSA with input $g(k, x)$ and output $x_s$.
11:                Set $l \leftarrow x_s$ and $\mathcal{F}_k \leftarrow 1$.
12:            until $\mathbf{f}(l) \npreceq \mathbf{f}(k)$.
13:            Append $x_k$ into the $OPF$.
14:        end if
15:    end if
16: end for
17: Output the $OPF$ and exit.
```

4) NON-DOMINATED QUANTUM OPTIMIZATION

The *Non-Dominated Quantum Optimization* (NDQO) algorithm [5] is formally stated in Alg. 3 and it constitutes an extension to the DHA algorithm conceived for multi-objective problems. The NDQO algorithm is based on the reference route-solution update process of the DHA. However, the unitary operator U_g [5], which is defined in Eq. (19), implementing the dominance operator is employed instead of the single lower-comparison operator U_{f_k}, which is defined in Eq. (20). Moreover, a vector $\mathcal{F}$ of binary flags is used for distinguishing whether a specific route has or has not been already processed. Initially, the NDQO algorithm opts for using the first route-solution of the route index as the reference route, which is the direct route. This is carried out by modifying its respective flag value to 1, i.e. by assigning $\mathcal{F}_1 \leftarrow 1$, and it then activates a BBHT-QSA process for searching for a route-solution that dominates the reference route. Should the BBHT-QSA successfully output a valid route-solution, the reference route is forwarded to the BBHT-QSA output, while simultaneously the output route-solution flag value is set to 1, and then a new BBHT-QSA process is activated. This process is repeated until an invalid route-solution is output by the BBHT-QSA, indicating that the reference route is itself a Pareto optimal and, hence, it is appended to the OPF set. We note that this process is referred to as the *BBHT-QSA chain*. Subsequently, the next route of the route-solutions list is checked as to whether it has already either been processed or been dominated by the hitherto generated OPF. If outcome of this check is false, a new BBHT-QSA chain is activated with this route being the reference route. Otherwise, the route is disregarded and the next route is checked. This procedure is repeated until all the legitimate routes have been checked, imposing a complexity quantified in terms of U_g activations, which is on the order of $O(N)$ and of $O(N\sqrt{N})$ in the best- and worst-case scenario, respectively [5]. More specifically, the lower and the upper bounds of the complexity imposed are equal to [5]:

$$L_{NDQO}^{\min} = 4.5\sqrt{N} + \log_{\lambda}\left(4.5\frac{\lambda-1}{m}\sqrt{N}+1\right) + N, \quad (23)$$

$$L_{NDQO}^{\max} = 9.5N\sqrt{N} + N\log_{\lambda}\left(\sqrt{N}\right) + 9.125N + 22.5\sqrt{N}, \quad (24)$$

where $L_{NDQO}^{\min}$ and $L_{NDQO}^{\max}$ correspond to the complexity imposed in the best- and worst-case scenario, respectively.

IV. DESIGN METHODOLOGY

A. PARALLEL ORACLE GATE DESIGN

To the best of our knowledge, no parallel activation scheme has been proposed for quantum unitary operators. Therefore, the quantum circuit of the U_g operator in [5] should activate serially the U_{f_k} operators and then calculate the intersection of their outcomes according to Eq. (19). Explicitly, an intersection operation of multiple inputs, such as that of Eq. (19), can be implemented in the QD using a quantum *Toffoli Gate* [41]. Bearing this observation in mind, a possible implementation of the U_g operator is shown in Fig. 6, where both the *Quantum Control Register* (QCR) and the *Quantum Index Register* (QIR) outputs of each U_{f_k} operators are fed forward to the next $U_{f_{k+1}}$ operator. Moreover, each U_{f_k} operator is provided with a single-qubit *Local Oracle Workspace* (LOW) register, where the outcome of each comparison is stored. Then, the states of all the LOW registers along with a *Global Oracle Workspace* (GOW) register – which is initialized to the $|t\rangle$ state – are fed into the Toffoli gate, which then performs the intersection operation. Explicitly, the n-qubit Toffoli gate T [62] has a transfer matrix of [41]:

$$T_n = \begin{bmatrix} I_{2^n-2} & \mathbf{0}_{2^n-2,1} & \mathbf{0}_{2^n-2,1} \\ \mathbf{0}_{1,2^n-2} & 0 & 1 \\ \mathbf{0}_{1,2^n-2} & 1 & 0 \end{bmatrix}, \quad (25)$$

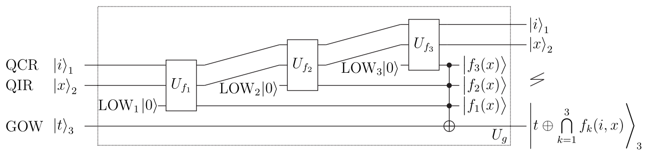


FIGURE 6. Serial implementation of the domincance operator used in NDQO for three optimization objectives.

where the $I_{2^{n-1}}$ sub-matrix denotes the a $[(2^n-2)\times(2^n-2)]$-element identity matrix and the vector $\mathbf{0}_{1,2^{n-1}}=\mathbf{0}^T_{2^n-2,1}$ contains (2^n-2) zero elements. Since in our scenario, each operator U_{f_k} has a single-qubit LOW output and the GOW is also comprised of a single qubit, we will use a 4-qubit Toffoli gate, i.e. we have $n=4$ qubits in Eq. (25).

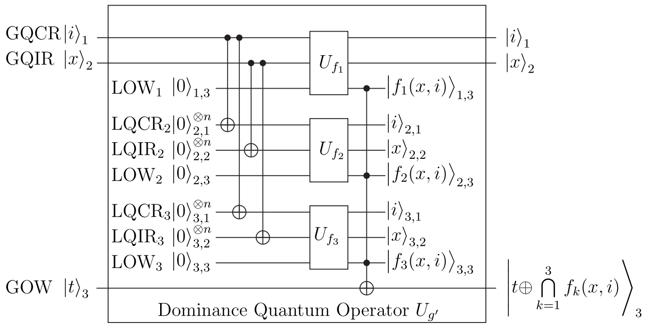

GQCR $|i\rangle_1$
GQIR $|x\rangle_2$
LOW$_1$ $|0\rangle_{1,3}$
LQCR$_2|0\rangle^{\otimes n}_{2,1}$
LQIR$_2$ $|0\rangle^{\otimes n}_{2,2}$
LOW$_2$ $|0\rangle_{2,3}$
LQCR$_3|0\rangle^{\otimes n}_{3,1}$
LQIR$_3$ $|0\rangle^{\otimes n}_{3,2}$
LOW$_3$ $|0\rangle_{3,3}$
GOW $|t\rangle_3$
U_{f_1}
U_{f_2}
U_{f_3}
$|i\rangle_1$
$|x\rangle_2$
$|f_1(x,i)\rangle_{1,3}$
$|i\rangle_{2,1}$
$|x\rangle_{2,2}$
$|f_2(x,i)\rangle_{2,3}$
$|i\rangle_{3,1}$
$|x\rangle_{3,2}$
$|f_3(x,i)\rangle_{3,3}$
$\left|t\oplus\bigcap_{k=1}^{3}f_k(x,i)\right\rangle_3$
Dominance Quantum Operator $U_{g'}$

FIGURE 7. Parallel implementation of the domincance operator used in NDQIO for three optimization objectives.

Moving on to the parallel implementation of the dominance operator, we propose the employment of the unitary operator $U_{g'}$, which relies on the quantum circuit implemented in Fig. 7. The main difference lies in the use of CNOT gates right before the input of the unitary operators U_{f_k} that implement the low-comparison-check, compared to that of Fig. 6, where the QIR and QCR outputs of the unitary operator U_{f_k} are fed forward to the $U_{f_{k+1}}$ unitary operator. These CNOT gates are invoked for entangling the states of both the *Global Quantum Control Register* (GQCR) and the *Global Quantum Index Register* (GQIR) with the states of respective local QRs for creating composite states. Their employment is essential for forming the composite state of:

$$|\psi\rangle^{out}_{LOW}=\sum_{x=0}^{N-1}|f_1(x,i)\rangle_{1,3}\,|f_2(x,i)\rangle_{2,3}\,|f_3(x,i)\rangle_{3,3} \qquad (26)$$

at the LOW outputs. In the absence of entanglement, the respective LOW output state will be:

$$\left|\psi'\right\rangle^{out}_{LOW}=\sum_{x_1=0}^{N-1}\sum_{x_2=0}^{N-1}\sum_{x_3=0}^{N-1}|f_1(x_1,i)\rangle_{1,3}\,|f_2(x_2,i)\rangle_{2,3}\,|f_3(x_3,i)\rangle_{3,3}, \qquad (27)$$

making the implementation of the dominance operator infeasible.

With the entanglement achieved by employing a series of CNOT gates, the $U_{g'}$ operator has an identical effect to that of the U_g operator, hence we have:

$$\sum_{x=0}^{N-1}|i\rangle_1\,|x\rangle_2\,|t\rangle_3 \xrightarrow{U_g,U_{g'}} \sum_{x=0}^{N-1}|i\rangle_1\,|x\rangle_2\,|t\oplus g(x,i)\rangle_3\,. \qquad (28)$$

Consequently, if we set the GOW register to the state $|0\rangle_3$, the $U_{g'}$ operator would return the outcome of the dominance operator and store it in the GOW output. Otherwise, if the GOW register is set to the state $|-\rangle_3$, then the $U_{g'}$ operator operates identically to the Quantum Oracle Gate O of Grover's QSA, i.e. by flipping the phase of those specific route-solutions, which are associated with $g(x,i)=1$. Furthermore, the main improvement of the new design relies on the fact that we have achieved the parallel activation of the unitary operators U_{f_k}. Therefore, should we assume that a single U_g activation corresponds to a single CFE both in terms of its execution time and of its power consumption, a single $U_{g'}$ operator activation would then correspond to a single CFE in terms of its power consumption and $1/3$[3] CFEs in terms of its execution time. We note that for the sake of simplicity, we have assumed that both the series of CNOT gates and the Toffoli gate consume negligible power compared to the U_{f_k} operators and that their response time is also negligible. Having presented our new parallel oracle design, let us now proceed with our detailed discussions of the NDQIO algorithm in the next section.

B. NON-DOMINATED QUANTUM ITERATIVE OPTIMIZATION

Our ultimate target is to reduce the lower bound of the complexity below the linear complexity dependence on the search-space size, yielding a further reduction of the average complexity. Therefore, we have revisited the framework of [5] with the objective of conceiving a hybrid design relying both on hardware parallelism and on quantum parallelism. More specifically, we will introduce a low-complexity initialization process for identifying the globally optimal routes, along with a sophisticated quantum-assisted process for finding new and potentially optimal routes. Furthermore, we have introduced an OPF *Self-Repair* (SR) process, which discards the suboptimal routes that have been erroneously included in the OPF, hence providing the NDQIO with an improved accuracy compared to the near-optimal NDQO algorithm's accuracy [5].

The NDQIO algorithm is formally stated in Alg. 4, where each distinct block is annotated using a comment starting with the character "#". In a nutshell, the NDQIO algorithm initializes the OPF to an empty set during Step 4.1 and then invokes the initialization process of Steps 4.3-6, where the DHA is activated as many times as the number of optimization objectives for the sake of identifying the globally optimal routes in terms of each objective. Subsequently, the

[3]Should K optimization objectives be considered, the execution time of the $U_{g'}$ operator will be equal to 1/K CFEs.

Algorithm 4 Non-Dominated Quantum Iterative Optimization Algorithm

```
1: Set $OPF \leftarrow \emptyset$.
2: # Initialization Process:
3: for $k = 1$ to $K$ do
4:     Invoke the DHA of Alg. 2 with input function $f_k(x, i)$,
       where $i$ is the index of a random legitimate route, and
       output $x_s$.
5:     Append $x_s$ to the $OPF$.
6: end for
7: # Iterative Step:
8: repeat
9:     # Backward BBHT-QSA Step:
10:    Set $\mathcal{F} \leftarrow 0$ and $\mathcal{T} \leftarrow 0$.
11:    repeat
12:        Invoke the BBHT of Alg. 1 with input $G(x, OPF)$
           and output $x_s$.
13:        if $G(x_s, OPF) = 1$ then
14:            Set $\mathcal{F} \leftarrow 2$ and $\mathcal{T} \leftarrow 1$.
15:        else
16:            Set $\mathcal{F} \leftarrow \mathcal{F} + 1$.
17:        end if
18:    until $\mathcal{F} = 2$
19:    if $\mathcal{T} = 1$ then
20:        # BBHT-QSA Chain:
21:        repeat
22:            Set $i \leftarrow x_s$.
23:            Define the oracle function $g(x, i)$ from (19).
24:            Invoke the BBHT-QSA of of Alg. 1 with input
               $g'(x, i)$ and output $x_s$.
25:        until $\mathbf{f}(x_s) \not\succeq \mathbf{f}(i)$.
26:        # Self-Repair Mechanism:
27:        Discard the routes from the OPF that are dominated
           by the $i$-th one.
28:        Append $i$ to the OPF.
29:    end if
30: until $\mathcal{T} = 0$.
31: Export the $OPF$ and exit.
```

iterative part of the algorithm is activated. At each iteration of Steps 4.8-30, the algorithm initially searches for a route, which is not dominated by the hitherto generated OPF using the BBHT-QSA process of Step 4.12. Should it succeed in identifying an appropriate route, it activates the BBHT-QSA chain of Steps 4.20-25, in the same fashion as the one in the NDQO algorithm. After the completion of the chain, the OPF Self-Repair (OPF-SR) process is invoked in Steps 4.27-28, where the routes of the OPF generated so far are checked to ascertain, whether they are dominated by the optimal route identified by the current iteration of the BBHT-QSA chain of Steps 4.20-25. The algorithm terminates and outputs the OPF, when the BBHT-QSA fails to identify a new potentially optimal route as formally defined by the condition of Step 4.30, concluding that there exists no other Pareto optimal route. Last but not least, we note that all the single-objective comparisons as well as the dominance operator activation have been carried out using the same quantum unitary operators as those used for forming the QSAs quantum oracles; the only difference relies upon the fact that in the OW register the input is initialized to the $|0\rangle$ state. Therefore, in constrast to the NDQO algorithm [5], in the NDQIO algorithm there is no distinction between the CD- and the QD-CFEs, since they are exclusively undertaken in the QD. Let us now proceed with our detailed discussions on each distinct sub-process of the NDQIO algorithm.

1) INITIALIZATION PROCESS

In the NDQO algorithm, which is formally stated in Alg. 3, no initialization process has been used; instead, the algorithm considers by default the first index of the legitimate route list as the first reference route and then initiates a BBHT-QSA chain. Despite the reduction offered by the hardware parallelization, the power consumption remains the same as that of the NDQO algorithm, as we demonstrated in Subsection IV-A. In fact, it is possible to achieve the same reduction in the power consumption as well by using the DHA for identifying the globally optimal routes in terms of each objective. To elaborate further, a single unitary operator U_{f_k}, which is defined in Eq. (20), is used for implementing a comparison in terms of the k-th objective, yielding a reduction in the power consumption per DHA activation, which is proportional to the number of optimization objectives. In fact, assuming K optimization objectives, this power consumption reduction results in identifying K OPF routes, while consuming the same amount of power and time as in a single NDQO BBHT-QSA chain, which would only identify a single OPF route. More explicitly, a single DHA activation imposes the same amount execution time compared as a single NDQO BBHT-QSA chain, while it simultaneously is also offering a power consumption reduction by a factor of $1/K$. We note that in our case study we have considered $K = 3$ optimization objectives according to Eq. (4). However, the application of the DHA is limited to identifying only globally optimal routes in terms of a single objective and not the Pareto optimal routes in general. Hence, the number of DHA activations is strictly limited to a maximum K routes. Additionally, we note that we have used an improved version of the DHA, which has been initially proposed in [47], and terminates the algorithm as soon as the BBHT-QSA fails to spot a legitimate solution, while exhausting its maximum affordable number of $\mathcal{G}$ applications.

2) SEEKING THE NEXT OPTIMAL ROUTE

After the completion of the initialization process we will acquire an OPF consisting of k OPF routes, where we have $k \in \{1, 2, \ldots, K\}$. The maximum value of $k = K$ corresponds to the case, where each objective is optimized by finding different routes, while the minimum value of $k = 1$ corresponds to the case, where only a single optimal route-solution exists, which is globally optimal for all the objectives considered. In the latter case, based on Def. 2, the true OPF will solely be comprised of this route. Nevertheless, since the BBHT-QSA and, inherently, the DHA exhibit a small but non-negligible probability of failing to identify a valid solution [29], the algorithm has to ensure that there are no unidentified Pareto optimal routes.

In the NDQIO algorithm, we have avoided the serial processing of the routes by employing a BBHT-QSA for finding the next potentially Pareto optimal route, hence achieving some complexity reduction. To elaborate further, our algorithm searches for route-solutions, which are not dominated by the OPF generated so far. For this reason, we can use our novel operator $U_{g'}$ for checking as to whether a route is or is not dominated by a reference route. This is realized by performing a swap between the states stored in the GQCR and the GQIR resulting in the state $|t \oplus g(i,x)\rangle_3$ at the GOW output of the $U_{g'}$ operator. In this case, we initialize the GOW input to the state $|t\rangle_3 \leftarrow |1\rangle_3$, while the respective GOW output becomes $|1 \oplus g(i,x)\rangle$, resulting in invoking the non-dominance operator. Explicitly, based on Eq. (19) the binary function $g(i,x)$ returns whether the i-th route does or does not dominate the x-th route and the operation $[1 \oplus g(i,x)]$ corresponds to the binary complement[4] of this function, implementing the non-dominance operator. Additionally, since the OPF is comprised of multiple routes, we have to use multiple $U_{g'}$ operators, each having a different OPF route as the reference route. Subsequently, using the novel framework presented in the Subsection IV-A, we can still achieve the parallel activation of the $U_{g'}$ operators by employing the series of CNOT gates for entangling the LQIRs state with the state of the GQIR at the input of the $U_{g'}$ operators along with a $(k+1)$-qubit Toffoli Gate, assuming having k reference routes, as portrayed in Fig. 8. As for the complexity imposed by a single activation of the U_G operator, it may be deemed to impose $1/k$ CFEs[5] in the execution time domain due to the parallel activation of the $U_{g'}$ unitary operators, which in turn activates the U_{f_k} unitary operators in parallel. In the power consumption domain, a single activation of the U_G operator imposes as many CFEs as the number of reference routes considered, which corresponds to the number of OPF routes that have been generated so far.

Moving on to the BBHT-QSA for identifying a specific route, which potentially belongs to the OPF, the U_G operator of Fig. 8 is used with its GOW register initialized to the state $|-\rangle_{k+2}$ and a BBHT-QSA is invoked with the OPF routes generated so far as reference ones, as stated in Step 4.4. By contrast, should the GOW register be initialized to the state $|0\rangle_{k+2}$, the operator U_G returns the non-dominance outcome at the output of the GOW register. Hence, we will utilize this initialization for performing the CD checks of the BBHT-QSA. Again, the BBHT-QSA exhibits a small but non-negligible probability of failing to identify a valid

[4]It returns whether the x-th route is dominated by the i-th one.

[5]We note that we have assumed that a single CFE corresponds to the activation of the serial unitary operator U_g of the NDQO algorithm.

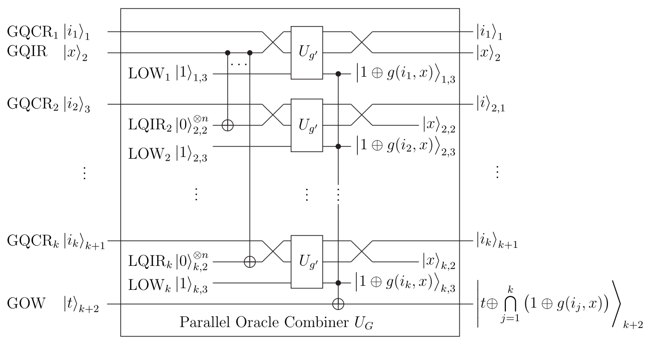

GQCR$_1$ $|i_1\rangle_1$
GQIR $|x\rangle_2$
LOW$_1$ $|1\rangle_{1,3}$
GQCR$_2$ $|i_2\rangle_3$
LQIR$_2$ $|0\rangle^{\otimes n}_{2,2}$
LOW$_2$ $|1\rangle_{2,3}$
GQCR$_k$ $|i_k\rangle_{k+1}$
LQIR$_k$ $|0\rangle^{\otimes n}_{k,2}$
LOW$_k$ $|1\rangle_{k,3}$
GOW $|t\rangle_{k+2}$
$U_{g'}$
$|1 \oplus g(i_1,x)\rangle_{1,3}$
$|x\rangle_{2,2}$
$|1 \oplus g(i_2,x)\rangle_{2,3}$
$|x\rangle_{k,2}$
$|1 \oplus g(i_k,x)\rangle_{k,3}$
Parallel Oracle Combiner U_G
$|i_1\rangle_1$
$|x\rangle_2$
$|i\rangle_{2,1}$
$|i_k\rangle_{k+1}$
$\left|t \oplus \bigcap_{j=1}^{k} (1 \oplus g(i_j,x))\right\rangle_{k+2}$

FIGURE 8. Quantum circuit of the BBHT-QSA unitary operator U_G used in the BBHT-QSA Oracle of Step 4.12 . Each activation of the U_G operator would impose 1/3 CFEs in execution time domain due to the parallel activation of the unitary operators $U_{g'}$ as well as the parallel activation of the U_{f_k} operators within each $U_{g'}$. In the power consumption domain, a single activation of the U_G imposes as many CFEs as the number of reference routes considered, i.e. the number OPF routes that have been so far generated.

solution [29], while exhausting the maximum number of $\mathcal{G}$ applications. We have mitigated this effect by repeating the BBHT-QSA process for one more additional iteration (Steps 4.16 and 4.18), for the sake of avoiding the premature termination of the NDQIO algorithm. Explicitly, an erroneous timeout would terminate unexpectedly the NDQIO algorithm, leading to its inability to identify the entire OPF. After identifying a potential OPF route, we may employ a BBHT-QSA chain (Steps 4.20-26) as in the NDQIO algorithm. The only difference lies in the employment of the $U_{g'}$ operator in the respective quantum Oracle gate, which provides the sub-process with a complexity reduction by a factor of $1/K$ in the execution time domain, albeit no reduction in the power consumption domain. Let us now proceed with the detailed description of the OPF-SR process.

3) SELF-REPAIR PROCESS

Searching for the next potential route guarantees that the exported route $x_{s,2}$ will not be dominated by the OPF generated so far, as ensured by the check performed in Step 4.13. Consequently, the route $x_{s,1}$ identified by the BBHT-QSA chain in conjunction with the initial reference route $x_{s,2}$ being as optimal, will not be dominated either based on Def. 1. However, the event when $x_{s,1}$ may dominate one or more routes of the OPF is not mutually excluded due to the dominance operator being non-commutative. Consequently, there may exist suboptimal routes that have been erroneously included into the OPF, owing to a BBHT-QSA failure. Hence, we may readily check whether there is any OPF route from the previous iterations, which is dominated by the identified OPF route of the current iteration, and discard it from the OPF. This check may be implemented using the U_{g^*} operator. The global registers ought to be initialized to:

$$|i\rangle_1 \leftarrow \left|x_{s,2}\right\rangle_1, \quad |x\rangle_2 \leftarrow \left|OPF_j\right\rangle_2, \quad |t\rangle_3 \leftarrow |0\rangle_3. \tag{29}$$

Then, we only have to observe the state of the GOW register output. This process is repeated for all the routes belonging

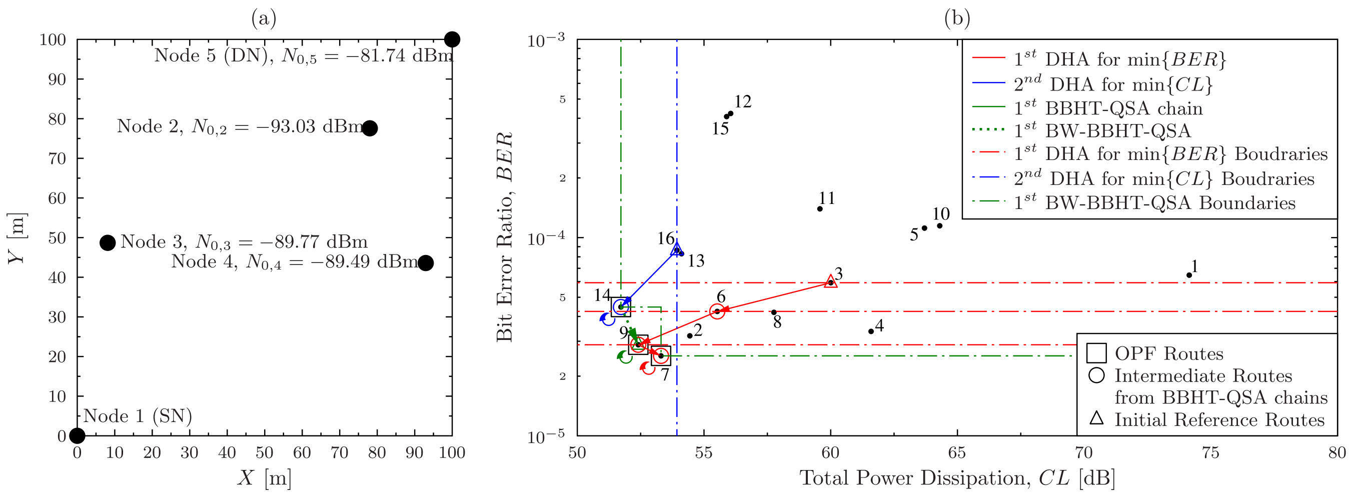


FIGURE 9. (a) Exemplified architecture for a 5-node WMHN, and (b) its optimization process using the NDQIO algorithm. In this example only two UFs are used per route-solution, for the sake of simplicity. The routes that belong to the OPF are marked by a square marker (□), the routes that initiate either a BBHT-QSA chain or a DHA activation are marked with a triangle (△), while the route-solutions output by each DHA or BBHT-QSA chain iteration, which are used as the new reference routes in the next iteration, are marked with a circle (○). Moreover, the indices of the routes as shown in Table 2 are marked in (b). Finally, the circural arrows in (b) denote that a BBHT-QSA has been activated with the respective route as its input, yet in the absence of potential route-solutions a random route is output by the BBHT-QSA, classifying the input route-solution as being Pareto Optimal (Step 4.25). The portrayed solution is not a unique one; different solutions could be derived depending on the DHA or BBHT-QSA chain's intermediate outcomes.

TABLE 2. Routes along with their UFs and indices for the exemplified 5-node WMHN of Fig. 9 [5].

Index i	Route	$P_{e,i}$ ($\times 10^{-4}$)	CL_i [dB]	Index i	Route	$P_{e,i}$($\times 10^{-4}$)	CL_i [dB]
1	{1 5}	0.646	74.147	9	{1 4 2 5}	0.288	52.407
2	{1 2 5}	0.319	54.440	10	{1 4 3 5}	1.147	64.302
3	{1 3 5}	0.592	60.004	11	{1 2 3 4 5}	1.397	59.575
4	{1 4 5}	0.336	61.593	12	{1 2 4 3 5}	4.230	56.053
5	{1 2 3 5}	1.117	63.700	13	{1 3 2 4 5}	0.829	54.113
6	{1 2 4 5}	0.424	55.524	14	{1 3 4 2 5}	0.446	51.721
7	{1 3 2 5}	0.253	53.304	15	{1 4 2 3 5}	4.079	55.893
8	{1 3 4 5}	0.420	57.759	16	{1 4 3 2 5}	0.863	53.931

to the OPF, as it was formally stated in the loop of Step 4.28. This repair process ensures that the NDQIO algorithm performs at its best attainable accuracy in terms the average Pareto distance $E[P_d]$, as long as the entire true OPF has been identified. Moving on to the consideration of the computational complexity imposed by the OPF-SR process, assuming that the multiple DHA activations provide us with k OPF routes and that the total number iterations carried out by the algorithm is equal to N_{OPF}, then in the first iteration we will have to invoke the $U_{g'}$ operator k times, while in the second iteration it will be activated $(k+1)$ times and so on, yielding a total number of CFEs that is equal to:

$$L_{SR}^{T} = \frac{1}{K}\sum_{i=k}^{N_{OPF}-1} i = \frac{1}{2K}(N_{OPF}^2 - k^2 - N_{OPF} + k), \quad (30)$$

$$L_{SR}^{P} = \sum_{i=k}^{N_{OPF}-1} i = \frac{1}{2}(N_{OPF}^2 - k^2 - N_{OPF} + k), \quad (31)$$

where L_{SR}^{T} and L_{SR}^{P} correspond to the complexity imposed in the execution time and the power consumption, respectively.

V. A DETAILED 5-NODE EXAMPLE

Having provided all the necessary discussions about the NDQIO algorithm's sub-processes, let us now provide an illustrative example for portraying the main concepts of our proposed algorithm. The exemplified WMHN structure is shown in Fig. 9(a), where the same 5-node WMHN structure is utilized as the one in [5]. Following a similar approach to that of [5], we will solely utilize two UFs for each route-solution, namely the BER and the CL, for facilitating the graphical representation of the route-solutions. Furthermore, the solution vectors and their respective indices, which correspond to all the legitimate routes, are presented in Table 2, while their graphical representation is shown in Fig. 9(b).

Additionally, we present all the necessary steps undertaken by the NDQIO algorithm in Fig. 9(b), where the route-solution transitions realized by the DHA or the BBHT-QSA chain activations are represented with the aid of arrows. Distinct colors have been used for representing the different sub-processes. In particular, as noted in the top legend of 9(b), the first and the second DHA activation transitions are annotated with red and blue arrows, respectively, while the BW-BBHT-QSA process and the BBHT-QSA chain transitions are indicated by green dashed and straight arrows, respectively. Moreover, the routes that belong to the OPF are marked by a square marker (□), the routes that initiate either a BBHT-QSA chain or a DHA activation are marked by a triangle (△), while the route-solutions output by each DHA or BBHT-QSA chain iteration, which are used as the new reference routes in the next iteration, are marked by a circle (○). Still referring to Fig. 9(b), the boundaries of the space, where valid route-solutions lie, are annotated by the long- and short-dashed lines. Let us now proceed with a more detailed description of the NDQIO algorithm's operation. We note that in terms of this tutorial example, the reader is assumed to be familiar with the concepts of the BBHT-QSA process.

Initially, the NDQIO algorithm sets the *OPF* set containing the optimal route-solution indices to an empty set, as formally stated in Step 4.1. Then, the algorithm's initialization process takes place (Steps 4.3–6), where the global minima are determined in terms of each optimization objective. To elaborate further, in this tutorial example we try to minimize both the *BER* and the *CL*, for the sake of simplicity, and, thus, we have $K = 2$ in Step 4.3. Therefore, we will activate the DHA process (Step 4.4) of Alg. 2 twice, i.e. once for each objective. Firstly, a DHA process is activated for minimizing the route's *BER*. More particularly, according to Step 2.1, a random route is chosen as the initial reference one and, then, a BBHT-QSA process is activated seeking a potential route-solution, which exhibits a lower *BER*. Let us assume that the initial reference route chosen is the one with index $i = 3$, i.e. the first DHA has chosen the route {1 3 5} according to Table 2, which is annotated with the red triangle (△) marker in Fig. 9(b). The arguments of the valid route-solutions of the BBHT-QSA process lie below the red horizontal long- and short-dashed line that crosses the argument of the route with index $i = 3$ in Fig. 9(b). Let us assume that the BBHT-QSA process outputs the route with index $i = 6$, i.e. the route {1 2 4 5} according to Table 2, which is marked with a red circle in Fig. 9(b).

Observe in Table 2 that the *BER* exhibited by the route with index $i = 6$ is lower than that exhibited by the reference route with index $i = 3$, i.e. we have $P_{e,6} < P_{e,3}$ and, thus, after the completion of the BBHT-QSA process invoked by the DHA in Step 2.5, the reference route will be updated to the BBHT-QSA output (Steps 2.7-8). Then, a new BBHT-QSA process is activated searching for a route with a lower *BER* that that of the reference one. Observe in Fig. 9(b) that the valid route-solutions lie below the red horizontal long- and short-dashed line that crosses the argument of the route with index $i = 6$. Therefore, the valid route-solution indices belong to the set $\{2, 4, 7, 8, 9\}$ and the BBHT-QSA process is capable of identifying any of them with equal probability. Let us assume that the output of the BBHT-QSA process is the route with index $i = 9$, i.e. we have $y' = 6$ in Step 2.5, and the reference route is updated, since we have $P_{e,9} < P_{e,6}$ according to Table 2. Still referring to Fig. 9(b), observe that the new reference route is indeed a Pareto optimal one. Nevertheless, the DHA process is unable to identify the route's property, since it is solely seeking a route with the minimum *BER*. Therefore, a new BBHT-QSA process is activated with the aid of the updated reference route, in which the only eligible output is the route with index $i = 7$, i.e. the route {1 3 2 5}. Hence, assuming that the BBHT-QSA process of Step 2.6 successfully identifies the latter route, the reference route is once again updated (Steps 27-8) and a new BBHT-QSA process is activated. Observe in Fig. 9(b) that the new reference route is indeed an optimal one in terms of its *BER*, hence, the BBHT-QSA will exhaust the maximum number of affordable $\mathcal{G}$ applications in the absence of valid solutions. Since in the design of the improved DHA of Alg. 2 we have set a single BBHT-QSA time-out as the termination condition (Steps 2.8, 12), the DHA exits and identifies the route associated with $i = 7$ as the optimal one in terms of its *BER* performance. Then, the NDQIO algorithm appends the DHA output to the *OPF* set in Step 4.5.

Then, a new DHA process is activated in search of the route, which is optimal in terms of *CL*. A new reference route is selected randomly among all the legitimate ones according to Step 2.1. At this stage, let us assume that the route associated with the index $i = 16$ is eventually selected, i.e. the route {1 4 3 2 5}, which is marked in Fig. 9(b) with the blue triangular marker. Then, a BBHT-QSA process is activated seeking a route exhibiting a lower *CL* than that of the reference route (Step 2.5). Observe in Fig. 9(b) that the valid route-solutions lie at the left-hand side of the vetrical blue long- and short-dashed line crossing the reference route and, in particular, the valid route-solutions have indices that belong to the set $\{7, 9, 14\}$. Assuming that the BBHT-QSA process outputs the route associated with the index $i = 14$, i.e. the route {1 3 4 2 5}, a new BBHT-QSA process is activated by updating the reference route, since the output route exhibits a lower *CL* than the reference one (Steps 2.7-8). The new reference route is the optimal one in terms of its *CL* performance, as portrayed in Fig. 9(b). Consequently, the BBHT-QSA process activated with this route being its input will exhaust the maximum number of affordable $\mathcal{G}$ applications, resulting in the input route's identification as the optimal one through Steps 2.7-12. After this operation, the DHA exits and outputs the identified optimal route, which is then incorporated into the OPF by the NDQIO algorithm (Step 4.5).

After the completion of the second[6] DHA, the initialization process ends and the iterative process (Steps 4.8-30) is activated. In the first part (Steps 4.10-18) of the NDQIO algorithm's iterative process, which is referred to as the *Backward BBHT-QSA Step* (BW-BBHT) in Alg. 4, a BBHT-QSA process is activated, which seeks a specific route-solution that is not dominated by the hitherto generated OPF and thus may potentially be a Pareto-optimal one. Explicitly, the arguments of the valid route-solutions lie in

[6]Assuming K optimization objectives, the initialization process ends right after the completion of the K-th DHA.

the area containing the center of the axes and bounded by the green long- and short-dashed lines, as portrayed in Fig. 9(b), where it is visible that the only eligible route-solution is the route associated with the index $i = 9$. We note that the BBHT-QSA process exhibits a slight probability of failing in terms of identifying a valid solution.[7] For this reason, this sub-process of the NDQIO algorithm has been designed to repeat the BBHT-QSA process in case of an unsuccessful search (Steps 4.15-18), which would prematurely terminate the NDQIO algorithm, hence substantially reducing the probability of an unsuccessful search. At this stage, let us assume that the BBHT-QSA process is able to identify the legitimate route-solution, which is no other than the route having the index $i = 9$, i.e. the route $\{1\ 4\ 2\ 5\}$.

Sequentially, the BBHT-QSA chain process of Steps 4.20-25 is activated in the same fashion as in the NDQO algorithm [5], with the reference route being the output of the BW-BBHT-QSA sub-process, which is the route with index $i = 9$. The BBHT-QSA chain process seeks a route-solution, which dominates the reference one. Nevertheless, observe in Fig. 9(b) that the initial reference route of the BBHT-QSA chain process is indeed a Pareto optimal route, i.e. there exists no route that dominates it. Therefore, the BBHT-QSA chain will activate only a single BBHT-QSA process, which will exhaust the maximum number of affordable $\mathcal{G}$ applications in the absence of valid route-solutions and hence will terminate the chain, according to condition of Step 4.25. Subsequently, the OPF self-repair sub-process of Step 4.27 is invoked, where the hitherto generated OPF routes are checked as to whether they are dominated by the Pareto optimal route-solution spotted by the BBHT-QSA chain. Should any of the route-solutions be dominated by the BBHT-QSA chain's output route-solution, they would be disregarded, since they would be suboptimal. In our example, the route-solution with index $i = 9$ does not dominate any of the already generated OPF routes according to Fig. 9(b) and thus the OPF remains intact.

After the completion of the OPF self-repair sub-process, the BBHT-QSA output route-solution is incorporated into the OPF and the iterative process of Steps 4.8-30 is repeated. Moreover, it is visible from Fig. 9(b) that the OPF, which is comprised of the routes-solutions associated with indices $\{7, 9, 14\}$ is identical to the TOPF. Hence, we conclude that the NDQIO algorithm has identified the entire TOPF. However, the NDQIO algorithm has no knowledge of this fact at this stage. By contrast, the BW-BBHT-QSA sub-process of Steps 4.10-18 is once again activated, where both the BBHT-QSA processes invoked by the NDQIO algorithm will exhaust the maximum number of $\mathcal{G}$ applications in the absence of valid route-solutions. This double time-out process would signal to the NDQIO algorithm that the entire OPF has been identified. Finally, the NDQIO algorithm exports the OPF identified and then exits. We note that we have made no mentioning of the NDQIO algorithm complexity, which is the subject of our discussions in the next section.

[7] This occurs when this solution is present in the examined database.

VI. COMPUTATIONAL ACCURACY VERSUS COMPLEXITY DISCUSSIONS

In this section, we will characterize our novel algorithm both in terms of its computational complexity and its accuracy. As benchmarking algorithms, we will employ the quantum-assisted NDQO algorithm of [5], and the Brute-Force (BF) method, since it was demonstrated in [5] that the NDQO outperforms both the NSGA-II and the ACO algorithm in terms of its accuracy at the complexity [5]. We note that the formal declaration of the BF method is shown in Alg. 5 [5]. Let us now proceed with quantifying the complexity imposed by the NDQIO algorithm.

Algorithm 5 Brute-Force Method [5]

```
1:  Initialize $OPF = \varnothing$.
2:  for $i = 0$ to $N - 1$ do
3:      Set $f \leftarrow 0$
4:      if $\nexists j \in OPF : \mathbf{f}(j) \succeq \mathbf{f}(i)$ then
5:          for $k = 0$ to $N - 1$ do
6:              if $\mathbf{f}(k) \succeq \mathbf{f}(i)$ then
7:                  Set $f \leftarrow 1$ and terminate inner loop.
8:              end if
9:          end for
10:         if $f = 0$ then
11:             Append $i$ into the $OPF$.
12:         end if
13:     end if
14: end for
15: Output the $OPF$ and exit.
```

A. NDQIO COMPLEXITY

In contrast to the NDQO algorithm [5], which is presented in Alg. 3, we will characterize the complexity imposed by the NDQIO algorithm, which is presented in Alg. 4, both in terms of its execution time and its power consumption. Since these parameters depend on the hardware used, we will normalize them to the specific execution time and the power consumption required for a single U_g operator, which is defined in Eq. (19), respectively. We note that due to this normalization process, the normalized execution time and power consumption of the NDQO will be identical. Hence, relying on the assumption that the operators $\{U_{f_k}\}_{k=1}^{K}$, defined in Eq. (20), impose the same execution time and power consumption irrespective of k, we may derive upper and the lower bounds of the NDQIO algorithm by examining both the worst- and the best-case scenarios considered in [5] for the NDQO algorithm, respectively. As far as the BF method is concerned, the execution time and the power consumption are identical in terms of CFEs, since hardware parallelism is not used. Hence, for these two extreme cases, the respective upper and lower bounds of the BF method are quantified in terms of CFEs as follows [5]:

$$L_{BF}^{\max} = N^2 + \sum_{i=0}^{N-1} i = \frac{3}{2}N^2 - \frac{1}{2}N = O(N^2), \quad (32)$$

$$L_{BF}^{\min} = 2N - 1 = O(N). \quad (33)$$

Let us now proceed with characterizing the NDQIO algorithm. For its lower bound, we will assume a scenario, where a single Pareto-Optimal route exists, namely the direct route and all the activated BBHT-QSA processes impose the minimum possible number of CFEs. Explicitly, they impose $L_{BBHT}^{\min}$ CFEs as defined in Eq. (21). Let us assume that during the initialization all the DHA processes, which are activated in Step 4.4, have their reference route initialized to the direct one in Step 2.1. Then, their first invoked BBHT-QSA process formulated in Alg. 1 will exhaust the maximum affordable number of $\mathcal{G}$ applications in the absence of valid route-solutions. This results in terminating the respective DHA process of Alg. 2, hence imposing a complexity of:

$$\begin{aligned} L_{NDQIO,init}^{\min} &= L_{BBHT}^{\min} + 1 \\ &= 4.5\sqrt{N} + \log_{\lambda}\left(4.5\frac{\lambda-1}{m}\sqrt{N}+1\right) + 2 \\ &= O(\sqrt{N}) \end{aligned} \quad (34)$$

in terms of the number of U_{f_k} activations, where the unitary operators U_{f_k} are defined in Eq. (20). We note that the first DHA process of Step 4.4, appends the direct route to the OPF. Therefore, if we take into consideration that the each U_{f_k} activation imposes $1/K$ CFEs in both domains and that K DHA processes are activated during the initialization process of Alg. 4, the complexity imposed by the NDQIO initialization process in both domains is equal to:

$$L_{NDQIO,init}^{T,\min} = L_{NDQIO,init}^{\min}, \quad (35)$$

$$L_{NDQIO,init}^{P,\min} = L_{NDQIO,init}^{\min}. \quad (36)$$

Subsequently, the iterative process of Alg. 4 is activated. Nevertheless, since there exist no routes that are not dominated by the direct one, the BBHT-QSA process of Alg. 1 that seeks a new potentially optimal route will reach its time-out twice and exit, hence imposing a complexity of:

$$\begin{aligned} L_{NDQIO,iter}^{\min} &= 2(L_{BBHT}^{\min} + 1) \\ &= 9\sqrt{N} + 2\log_{\lambda}\left(4.5\frac{\lambda-1}{m}\sqrt{N}+1\right) + 4 \\ &= O(\sqrt{N}) \end{aligned} \quad (37)$$

in terms of $U_{g'}$ activations. Thus, the respective execution time and power consumption imposed by the NDQIO iterative process becomes:

$$L_{NDQIO,iter}^{T,\min} = \frac{1}{K}L_{NDQIO,iter}^{\min}, \quad (38)$$

$$L_{NDQIO,iter}^{P,\min} = L_{NDQIO,iter}^{\min}. \quad (39)$$

The SR process of Steps 4.27-28 will not be activated, since no OPF route has been identified by the the BBHT-QSA process, which is invoked at Step 4.12. Therefore, the total execution time imposed by the NDQIO algorithm is derived by adding up Eqs. (35) as well as (38) and the result is shown in (50), as shown at the top of the next page. Equivalently, the total power consumption imposed by the NDQIO algorithm is derived by adding up Eqs. (36) as well as (39) and the result is shown in (51), as shown at the top of the next page. Observe in Eqs. (50) and (51) that the minimum execution time and power consumption imposed by the NDQIO algorithm in both domains is on the order of $O(\sqrt{N})$.

As for the upper bound, we will consider the case, where all the routes are Pareto-Optimal and the BBHT-QSA processes impose the maximum possible complexity of $L_{BBHT}^{\max}$, as quantified in Eq. (22). Under this assumption, each DHA process imposes the maximum possible number of U_{f_k} activations, when it activates precisely five BBHT-QSA chains, since we have:

$$4L_{BBHT}^{QD,\max} < 22.5\sqrt{N} < 5L_{BBHT}^{QD,\max}, \quad (40)$$

where $L_{BBHT}^{QD,\max} = 5.5\sqrt{N} - 1$ [5] corresponds to the maximum QD complexity in terms of U_{f_k} activations imposed by a single BBHT-QSA activation. Therefore, the maximum complexity $L_{DHA}^{\max}$ imposed by the DHA is equal to:

$$L_{DHA}^{\max} = 5(L_{BBHT} + 1) = 50\sqrt{N} + 5\log_{\lambda}\left(\sqrt{N}\right). \quad (41)$$

Consequently, the maximum execution time and power consumption imposed by the initialization process is equal to:

$$L_{NDQIO,init}^{T,\max} = K\frac{1}{K}L_{DHA}^{\max} = 50\sqrt{N} + 5\log_{\lambda}\left(\sqrt{N}\right), \quad (42)$$

$$L_{NDQIO,init}^{P,\max} = K\frac{1}{K}L_{DHA}^{\max} = 50\sqrt{N} + 5\log_{\lambda}\left(\sqrt{N}\right), \quad (43)$$

which was quantified as a function of the number of CFEs, resulting in an OPF constituted by exactly K routes. Moving on to the NDQIO iterative process (Steps 4.8-30), we will assume that the BBHT-QSA searching for a new potentially optimal route, fails during the initial activation but succeeds during the second one in identifying a valid route-solution. Furthermore, since all the routes are optimal, the BBHT-QSA chain will activate a single BBHT-QSA, which in turn exhausts the maximum affordable number of QD-CFEs, in the absence of valid solutions. Therefore, during each iteration precisely 3 BBHT-QSA processes will be activated. Explicitly, the complexity-dependent power consumption imposed by the BBHT-QSA that seeks a new potential route increases as the number of iterations increases, which is a consequence of increasing in the number of OPF routes used as reference routes. Hence, the maximum total time execution and power consumption imposed by this process of Steps 4.10-19 is equal to:

$$\begin{aligned} L_{NDQIO,BW}^{T,\max} &= \frac{1}{K}\sum_{k=K}^{N}\left[2(L_{BBHT}^{\max}+1)\right] \\ &= \frac{N-K}{K}\left[10\sqrt{N} + \log\lambda(\sqrt{N})\right], \end{aligned} \quad (44)$$

$$L^{T,\min}_{NDQIO,tot} = \frac{K+2}{K}\left[4.5\sqrt{N}+\log_\lambda\left(4.5\,\frac{\lambda-1}{m}\,\sqrt{N}+1\right)+2\right] = O(\sqrt{N}), \quad (50)$$

$$L^{P,\min}_{NDQIO,tot} = 3\left[4.5\sqrt{N}+\log_\lambda\left(4.5\,\frac{\lambda-1}{m}\,\sqrt{N}+1\right)+2\right] = O(\sqrt{N}), \quad (51)$$

$$L^{T,\max}_{NDQIO,tot} = \frac{2N+3K-1}{K}\left[10\sqrt{N}+\log_\lambda\left(\sqrt{N}\right)\right]+\frac{N^2-K^2-N+K}{2K} = O(N^2), \quad (52)$$

$$L^{P,\max}_{NDQIO,tot} = \left(N^2-K^2+N-K+4\right)\left[10\sqrt{N}+\log_\lambda\left(\sqrt{N}\right)\right]+\frac{N^2-K^2-N+K}{2} = O(N^2\sqrt{N}). \quad (53)$$

$$L^{P,\max}_{NDQIO,BW} = \sum_{k=K}^{N}\left[2k(L^{\max}_{BBHT}+1)\right] = \left(N^2-K^2\right)\left[10\sqrt{N}+\log\lambda(\sqrt{N})\right], \quad (45)$$

as a function of the number of CFEs, respectively, while that imposed by the BBHT-QSA chains of Steps 421-25 is equal to:

$$L^{T,\max}_{NDQIO,chain} = \frac{1}{K}\sum_{k=K}^{N-1}(L^{\max}_{BBHT}+1) = \frac{N-K-1}{K}\left[10\sqrt{N}+\log\lambda(\sqrt{N})\right], \quad (46)$$

$$L^{P,\max}_{NDQIO,chain} = \sum_{k=K}^{N-1}(L^{\max}_{BBHT}+1) = (N-K-1)\left[10\sqrt{N}+\log\lambda(\sqrt{N})\right], \quad (47)$$

respectively. Subsequently, the OPF-SR process of Steps 4.25-39 will be activated precisely $(N-K)$ times, imposing the execution time and the power consumption quantified in Eqs. (30) and (31), respectively, upon substituting $N_{OPF}=N$ and $k=K$, hence we have:

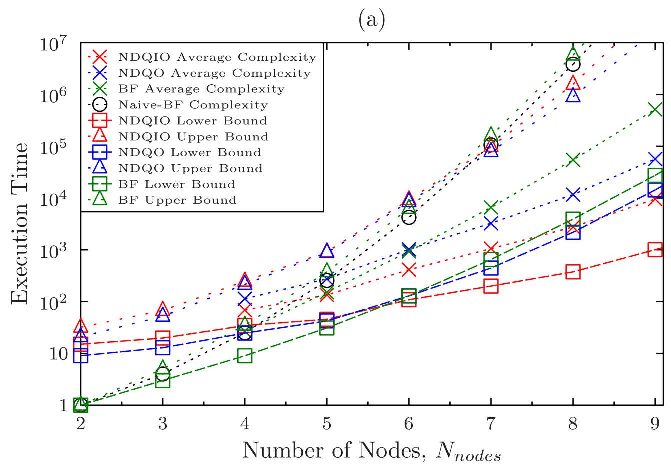
$$L^{T,\max}_{NDQIO,SR} = \frac{1}{K}\sum_{i=K}^{N-1} i = \frac{1}{2K}(N^2-K^2-N+K), \quad (48)$$

$$L^{P,\max}_{NDQIO,SR} = \sum_{i=K}^{N-1} i = \frac{1}{2}(N^2-K^2-N+K), \quad (49)$$

respectively. Following a similar approach to the lower bound derivation, the execution time upper bound imposed by the NDQIO algorithm is derived by adding together Eqs. (42), (44), (46) and (48) and the result of their addition is shown in Eq. (52), as shown at the top of this page. Equivalently, the power consumption upper bound is given by Eq. (53), as shown at the top of this page, stemming from the addition of Eqs (43), (45), (47) and (49). Observe in Eqs. (52) and (53) that the resultant execution time and power consumption upper bounds are on the order of $O(N^2)$ and $O(N^2\sqrt{N})$, respectively, which are higher than $O(N\sqrt{N})$ imposed by the NDQO algorithm [5].

Explicitly, this additional cost is justified by the increased elitism introduced by the NDQIO algorithm compared to the NDQO algorithm. To elaborate further, the NDQIO algorithm is capable of curtailing its operation upon detecting that there are no unidentified OPF routes. In the worst-case scenario, this imposes complexities on the orders of $O(N\sqrt{N})$ and $O(N^2\sqrt{N})$ in the execution time and the power consumption domains, respectively. By contrast, in the best case-scenario, the lower bound is on the order of $O(\sqrt{N})$ in both domains. Additionally, the OPF-SR process imposes a complexity on the order of $O(N^2)$ in both domains in the worst-case scenario, while no complexity is imposed in the best-case situation. Therefore, the complexity reduction achieved by the NDQIO algorithm is inherently related to the ratio of the total number N_{OPF} of the OPF routes over the total number N of the legitimate routes considered.

The average complexities of the NDQIO, of the NDQO algorithms and of the BF method along with their respective upper and lower bounds in the execution time and in the power consumption domains are shown in Figs. 10(a) and 10(b), respectively, for WMHNs consisting of $N_{nodes}=2$ to $N_{nodes}=9$. We note that the respective complexities of the NDQO algorithm and of the BF method will be identical in both domains, since they do not rely on any hardware parallelism techniques. As far as the upper bounds of the execution time are concerned, observe in Fig. 10(a) that the NDQIO algorithm involves the same order of complexity as the NDQO algorithm, when considering $N_{nodes}=2$ to $N_{nodes}=7$ nodes. This is justified by the fact that for WMHNs having more than $N_{nodes}=7$ nodes the term predominantly governing the respective complexity is the N^2 term, while for less densely populated WMHNs the upper bound is governed by the term $\frac{20}{K}N\sqrt{N}$ based on Eq. (52). Hence, the order of the execution time upper bound of the NDQIO algorithm for $N_{nodes}<7$ nodes is reduced to $O(N\sqrt{N})$, matching the order of the NDQO algorithm's upper bound. We note that in our case study we have assumed three optimization objectives, i.e. we have $K=3$. Moreover, for WMHNs consisting of more than $N_{nodes}=7$ nodes, we observe in Fig. 10(a) that the NDQIO upper bound is lower than those of both the naive-BF and of the BF methods due to the complexity reduction by a factor of $1/K$ offered by the hardware parallelism due to the employment of the $U_{g'}$ operator. On the other hand, the NDQIO upper bound of the power consumption is definitely governed by the term $10N^2\sqrt{N}$. Consequently, observe in Fig. 10(b) that it involves a several orders of magnitude higher power consumption than that of the benchmarking algorithms used for WMHNs



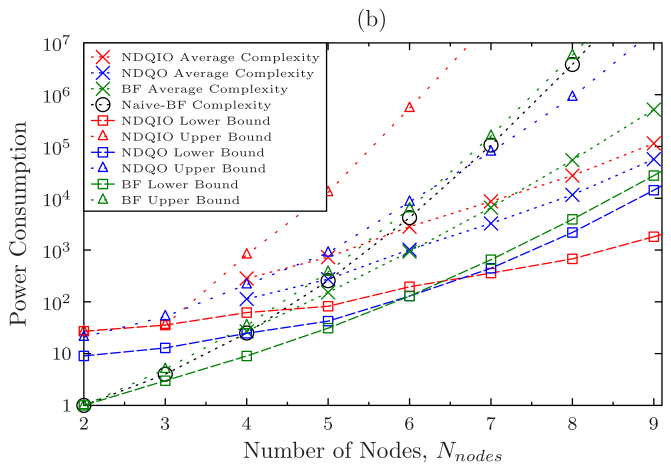


FIGURE 10. Evolution of (a) execution time and (b) power consumption of the NDQIO algorithm compared to the respective values imposed by the BF method of Alg. 5 and the NDQO algorithm. The mean complexity results have been averaged over 10^8 runs.

having more than $N_{nodes} = 3$ nodes. We note that for WMHNs having either $N_{nodes} = 3$ or $N_{nodes} = 2$ nodes, the NDQIO upper bound matches its lower bound, since in these cases the total number of routes is less than that of the number of objectives, which makes the iterative process unnecessary.

As for the NDQIO algorithm's execution time lower bound, observe in Fig. 10(a) that the NDQIO algorithm provides some complexity reduction compared to the benchmarking algorithms for WMHNs having $N_{nodes} = 6$. Explicitly, based on Eqs. (23) and (50), the NDQIO algorithm begins to outperform the NDQO, when the total number of routes is higher than $N_{routes} = 22$ routes, yielding that the this reduction becomes visible for WMHNs having $N_{nodes} = 6$, where the total number of legitimate route is equal to $N = 65$. In the power consumption domain, some complexity reduction is achieved for $N_{routes} > 134$ routes, corresponding to 7-node WMHNs, as shown in Fig. 10(b). Additionally, the NDQIO lower bound indicates a complexity reduction of several orders of magnitude, as demonstrated in Figs. 10(a, b).

Moving on to the average complexity in terms of the execution time, observe in Fig. 10(a) that the NDQIO algorithm outperforms the NDQO for the WMHN sizes considered. In particular, for WMHNs having $N_{nodes} = 4$ and $N_{nodes} = 5$ the execution time imposed by the NDQO algorithm is almost twice as high as that of the NDQIO, since the latter benefits from the hardware parallelism design. However, specifically for 4-node WMHNs it imposes a higher complexity than that of the BF method, since the NDQIO does not benefit from the complexity reduction offered by the QP for small search-spaces [5], solely relying on the complexity reduction offered by the parallel oracle design. As soon as the complexity reduction of the QP becomes significant, which occurs for WMHNs associated with $N_{nodes} \geq 6$ nodes, the average complexity imposed by the NDQIO algorithm becomes several orders of magnitude lower than that of the NDQO algorithm and that of the BF method, as portrayed in Fig. 10(a). We note that this complexity reduction becomes more significant as the number of nodes increases, since the complexity reduction offered by the quantum algorithms is improved as the search-space is increased [5], [29], [45]. As for the NDQIO algorithm's average power consumption, observe in Fig. 10(b) that it outperforms the BF method for WMHNs having $N_{nodes} = 8$ or more, while in the special case, where we have $N_{nodes} = 7$ the two algorithms impose a complexity on the same order. Compared to the NDQO algorithm, the NDQIO imposes about 2.5 times the respective complexity of the NDQO algorithm for WMHNs having four to seven nodes, while for more nodes the complexity imposed decays to about twice that of the NDQO algorithm.

Therefore, we conclude that both the NDQO and the NDQIO exhibit about the same order of power consumption, while at the same time the NDQIO algorithm offers a substantial execution time reduction of several orders of magnitude. This observation unveils a trade-off. Explicitly, based on Figs. 10(a, b) the NDQIO algorithm offers about ten times lower complexity at the expense of consuming twice the power compared to the NDQO, for WMHNs having eight or more nodes. Explicitly, this additional 100% power consumption overhead stems from the escalating number of OPF routes included in the BW-BBHT-QSA process of Step 4.12. Hence, every BW-BBHT-QSA iteration requires more power, due to the inclusion of more reference routes, albeit this is achieved without increasing the execution time required, owing to the parallel activation of the U'_g operators. However, we expect this 100% in power consumption overhead to gradually diminish as the ratio of the number of OPF routes over the total number of routes decreases due to the WMHN becoming more densely populated by nodes. This trend can be inferred from Figs. 10(a, b), by observing that both the average execution time and the average power consumption exhibit a larger distance from their respective upper bounds, as the number of nodes in the WMHN increases.

$$L^T_{NDQIO} = \frac{1}{2K}N^2_{OPF} + \frac{1}{K}\left(L_{DHA} + 2L_{BBHT} - \frac{1}{2}\right)N_{OPF} + (1-K)\left[\frac{2}{K}L_{BBHT} + \frac{1}{2}\right], \tag{54}$$

$$L^P_{NDQIO} = \left(\frac{1}{2} + L_{BBHT}\right)N^2_{OPF} + \left(L_{DHA} + L_{BBHT} - \frac{1}{2}\right)N_{OPF} + (1-K)\left(L_{DHA} + L_{BBHT} + \frac{1}{2}K\right). \tag{55}$$

Furthermore, we may observe in Fig. 10(b) that the average power consumptions of the NDQIO and NDQO algorithms become similar, as the number of nodes associated with the WMHN increases. Therefore, a critical question arises, whether the NDQIO algorithm would asymptotically approach the average power consumption of the NDQO algorithm or whether it would outperform it. For this reason, let us codunct a further case study, where we will determine both the normalized execution time and the power consumption of the algorithms in terms of the number N_{OPF} of the optimal routes belonging to the OPF.

As far as the NDQO algorithm is concerned, its BBHT-QSA chains impose a complexity identical to that of the DHA, since the BBHT-QSA chain constitutes an extension of the DHA for multi-objective problems. Since precisely N_{OPF} BBHT-QSA chain processes will take place, the resultant complexity imposed by this process is equal to:

$$L_{NDQO,chain} = N_{OPF}L_{DHA}. \tag{56}$$

For the serial parsing step, let us stipulate the further assumtion that the routes initiating a BBHT-QSA chain are distributed uniformly within the route-database and that - on average - the routes require a dominance comparison with half the routes of the hitherto generated OPF. Consequently, a BBHT-QSA chain is invoked for every N/N_{OPF} routes and the resultant complexity becomes:

$$L_{NDQO,sp} = \frac{N}{N_{OPF}}\sum_{i=1}^{N_{OPF}}\frac{i}{2} = \frac{N(N_{OPF}+1)}{4}. \tag{57}$$

Hence, the overall complexity imposed by the NDQO algorithm is derived by adding up Eqs. (56) and (57) yielding:

$$L_{NDQO} = N_{OPF}\left(L_{DHA} + \frac{N}{4}\right) + \frac{N}{4}. \tag{58}$$

Since the DHA and inherently the BBHT-QSA chain impose a complexity on the order of $O(\sqrt{N})$, the NDQO complexity will be on the order of $O(N_{OPF}N)$, based on Eq. (58). As for the algorithm's normalized execution time and power consumption, they will be identical to the complexity imposed, since the NDQO algorithm does not involve hardware paralellism.

Let us now proceed by characterizing the average normalized time execution and power consumption imposed by the NDQIO algorithm. The NDQIO algorithm invokes the DHA K times, once per objective, plus $(N_{OPF} - K)$ times a BBHT-QSA chain for the rest of the optimal routes, yielding a normalized execution time and a power consuption of:

$$L^T_{NDQIO,chain} = \frac{N_{OPF}}{K}L_{DHA}, \tag{59}$$

$$L^P_{NDQIO,chain} = (N_{OPF} + 1 - K)\,L_{DHA}, \tag{60}$$

respectively. We note that the terms $L^T_{NDQIO,chain}$ and $L^P_{NDQIO,chain}$ consider the execution time and the power consumption, respectively, which are imposed by both the BBHT-QSA chain and the initialization process. Additionally, the execution time and the power consumption imposed by these processes is on the order of $O(N_{OPF}\sqrt{N})$. Subsequently, let us consider the worst-case scenario for the backward-oriented BBHT-QSA process, where two BBHT-QSA search processes are activated, yielding an unsuccessful output from the first, whilst the second succeeds in identifying a potentially optimal route. This process is activated $(N_{OPF} - K + 1)$ times, resulting in an execution time and a power consumption equal to:

$$L^T_{NDQIO,BW} = \frac{1}{K}\sum_{i=K}^{N_{OPF}}(2L_{BBHT}), \\ = \frac{2\,(N_{OPF} - K + 1)}{K}L_{BBHT}, \tag{61}$$

$$L^P_{NDQIO,BW} = \sum_{i=K}^{N_{OPF}}(2iL_{BBHT}), \\ = \left(N^2_{OPF} - K^2 + K + N_{OPF}\right)L_{BBHT}, \tag{62}$$

respectively. Therefore, since the BBHT-QSA complexity is on the order of $O(\sqrt{N})$, the normalized execution time and the power consumption of the backward BBHT-QSA process is on the order of $O(N_{OPF}\sqrt{N})$ and $O(N^2_{OPF}\sqrt{N})$, respectively. As for the SR process, the execution time and power consumption imposed have been derived in Eqs. (30) and (31), respectively. Hence, the overall execution time imposed by the NDQIO algorithm is derived in Eq. (54), as shown at the top of this page, by adding together Eqs. (30), (59) and (61). Similarly, the overall power consumption is shown in Eq. (55), as shown at the top of this page, as a result of the addition of Eqs. (31), (60) and (62).

Observe in Eqs. (54) that the amount of the overall normalized execution time and that of the normalized power consumption imposed by the NDQIO algorithm are on the order of $O(N_{OPF}\sqrt{N})$ and of $O(N^2_{OPF}\sqrt{N})$, respectively, as opposed to those imposed by the NDQO, which are both on the order of $O(N_{OPF}N)$. Hence, a further investigation on the order of the number N_{OPF} of optimal routes in terms of the total number N of legitimate should be carried out. For this reason, let us assume that all the QD processes impose the maximum possible complexity, i.e. we set $L_{DHA} = L^{\max}_{DHA}$

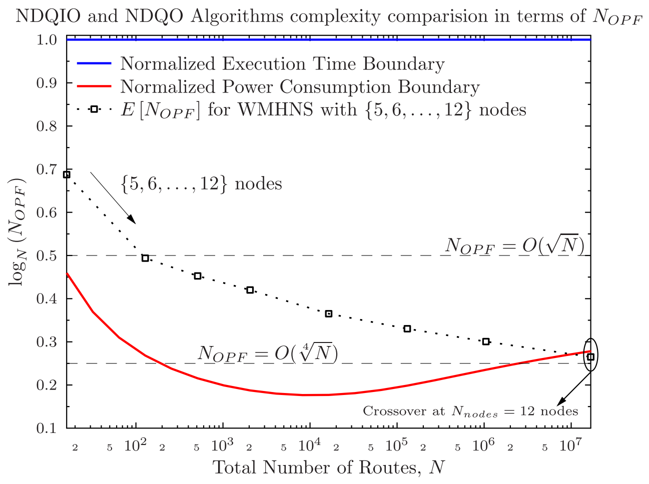


FIGURE 11. **Normalized execution time and power consumption boundaries, where the NDQIO algorithm outperforms the NDQO one. The average number N_{OPF} of optimal routes based on our simulation setup is portrayed with the black squares WMHNs consisting of 5 until 12 nodes. The results have been averaged over 10^8 runs.**

and $L_{BBHT} = L_{BBHT}^{\max}$, as defined in Eqs. (41) as well as (22) and investigate the number of optimal routes for which the NDQIO algorithm succeds in outperforming the NDQO one, where we have:

$$L_{NDQO} > L_{NDQIO}^{T}, \tag{63}$$

$$L_{NDQO} > L_{NDQIO}^{P}, \tag{64}$$

for the normalized execution time and for the power consumption, respectively. The solution of Eq. (63) for the normalized time execution time is portrayed in Fig. 11 using a blue line. Observe that the boundary is constant and equal to unity, demonstrating that the condition of Eq. (63) is satisfied for every N_{OPF} in the range of $1 \leq N_{OPF} \leq N$. Hence, the NDQIO algorithm outperforms the NDQO in terms of their normalized power consumtpion regardless of the number N of the WMHN nodes and of the number N_{OPF} of Pareto-optimal routes, which is verified in Fig. 10(a). On the other hand, the corresponding normalized power consumption boundary is shown in Fig. 11 by the red line. Hence, the latter is then compared to the average number $E[N_{OPF}]$ of Pareto-optimal routes, which were exported from our WMHN routing problem for WMHNs having between five and twelve nodes, in order to ascertain whether they are lower than the respective bound. In fact, observe in Fig. 11 that the average number of optimal routes lies above that corresponding to the power consumption bound for WMHNs having up to 11 nodes. However, there is a crossover in Fig. 11 between the normalized power consumption and the average number of OPF routes for the 12 nodes, indicating that the NDQIO algorithm will eventually outperform the NDQO in terms of the normalized power consumption.

B. NDQIO COMPUTATIONAL ACCURACY PERFORMANCE

Having characterized the NDQIO algorithm in terms of its complexity imposed and its power consumption, let us now examine the algorithm's performance in terms of its Average Pareto Distance $E[P_d]$ and the Average Pareto Completion Ratio $E[C]$. Assuming that the optimal route indices form the set OPF, the aforementioned accuracy metrics are defined as [5]:

$$E[P_d] = \sum_{x\in OPF} \frac{P_d(x)}{|OPF|}, \tag{65}$$

$$E[C] = E\left[\frac{|OPF| - |OPF_e|}{|TOPF|}\right], \tag{66}$$

where the set OPF_e contains the suboptimal routes contained in the OPF set, the set $TOPF$ contains the truly optimal routes and $P_d(x)$ corresponds to the Pareto distance of the x-th route as defined in Eq. (5). The latter metric corresponds to the specific portion of the TOPF identified by the respective optimization method.

Let us now describe the evaluation process used for the NDQIO algorithm. When using a similar approach to that involved for the NDQO algorithm, the iterative process does not necessarily impose the same number of CFEs, due to the stochastic nature of the BBHT-QSA [29]. Hence the evaluation process will be invoked each time a route is appended to the OPF. This event occurs right after the initialization process and after the completion of the iterative process, i.e. right after Steps 4.6 and 4.33, respectively. However, since the total number of CFEs required by both of the DHAs of the initialization process, the BBHT-QSAs of the iterative process are rather random processes, the evaluation process will be activated at different complexity values. We will assume that between these evaluation processes the metrics remain constant, which results in a sum of step functions for each simulation. We can then extract a continuous distribution for these metrics versus the number of CFEs by performing an averaging operation in each respective domain.

These metrics are shown in Fig. 12 for 7-node WMHNs. As far as the average Pareto distance $E[P_d]$ is concerned, observe in Figs. 12(a, b) that the NDQO algorithm performs optimally for 502 CFEs in both complexity and power consumption terms. It also exhibits an almost constant average Pareto distance $E[P_d]$, which is on the order of 10^{-8}. By contrast, the NDQIO algorithm exhibits an initial average Pareto distance $E[P_d]$, which is on the order of 10^-4 and is then reduced as the number of the iterative NDQIO steps increases. To elaborate further, as the NDQIO algorithm invests in more CFEs, i.e. more iterative steps, the number of identified OPF routes increases and as the self-repair process is invoked at the end of each iterative step, the probability of identifying an optimal route that dominates a suboptimal one erroneously included in the OPF increases. Consequently, the average Pareto distance $E[P_d]$ drops as the number of CFEs increases, as portrayed in Fig. 12(a, b). Additionally, observe in Fig. 12(a) that the NDQIO algorithm outperforms the NDQO one after about 1672 CFEs in the execution time domain and then after a total of 2008 CFEs the $E[P_d]$ becomes equal to zero, providing our algorithm with optimal performance in terms of this metric. The same holds for the power consumption, where the NDQIO algorithm outperforms the NDQO one

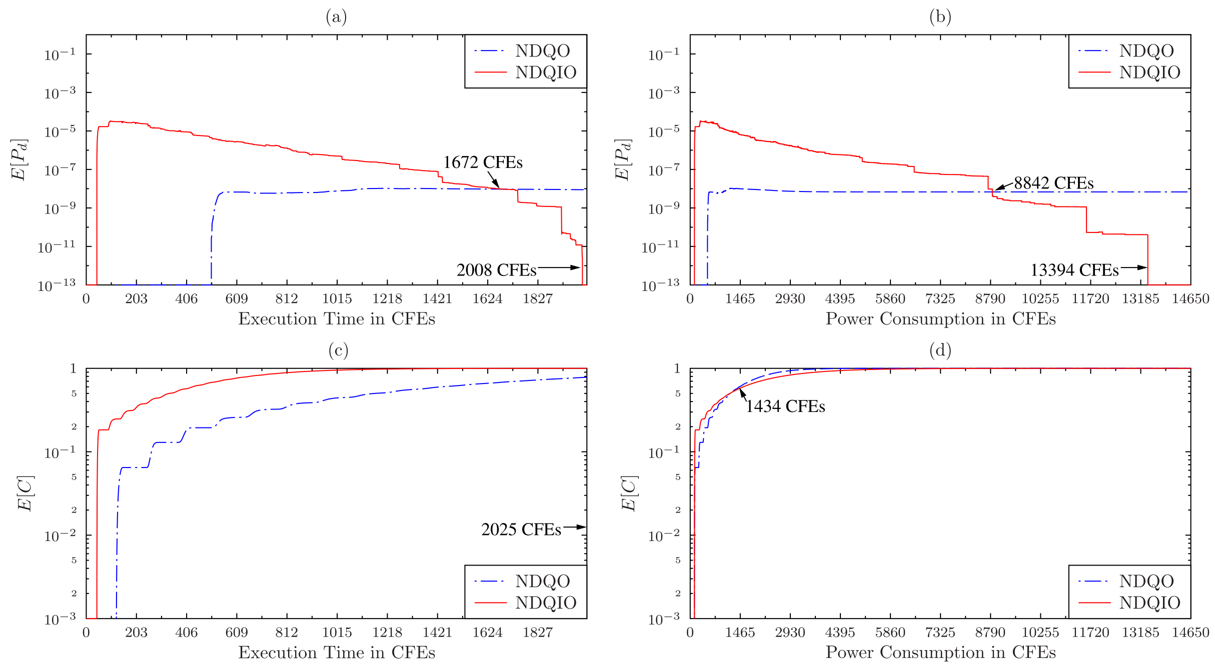


FIGURE 12. Perfomance comparison between the NDQIO and the NDQO algorithms and for 7-node WMHNs in terms of the Average Pareto Distance $E[P_d]$ (a, b) and Optimal Pareto Front Completion Ratio $E[C]$ (c, d) in vesus the execution time (a, c) and the power consumption (b, d). The results have been averaged over 10^8 runs.

after about 8753 CFEs and then after a total of 13394 CFEs the $E[P_d]$ becomes equal to zero, as portrayed in Fig 12(b).

Moving on to the NDQIO algorithm's performance appraised in terms of the average Pareto Completion Ratio $E[C]$, this is portrayed in Figs. 12(c, d). As far as the execution time is concerned, observe in Fig. 12(c) the NDQIO algorithm's completion probability converges to unity after 2025 CFEs, as opposed the 6491 CFEs imposed by the NDQO algorithm, yielding a further complexity reduction of about 68.80%. By contrast, in the power consumption domain, the NDQIO algorithm achieves a completion probability of unity after 14651 CFEs, which imposes an additional power consumption of 125.71%, compared to the NDQO algorithm. Moreover, observe in Fig. 12(d) that the NDQIO requires fewer CFEs to produce the first OPF routes in terms of the minimum consumption than the NDQO algorithm. This is a benefit of using the DHAs in the initialization process, which are capable of identifying as many OPF routes as the number of the optimization objectives, hence imposing an overall complexity equal to a single BBHT-QSA chain in the NDQO, which is capable of identifying a single OPF route. However, as the number of OPF routes identified increases, the power consumption of the BBHT-QSA seeking new potential OPF routes increases. Explicitly, the NDQO algorithm becomes more efficient after about 1434 CFEs, as shown in Fig. 12(d).

VII. CONCLUSIONS

We have proposed a novel hardware parallelization framework for quantum processes, which offers a further complexity reduction in addition to that provided by QP. Based on this framework, we have developed a novel algorithm as an improvement of the existing NDQO algorithm. As a benefit of the hardware parallelization, we have distinguished the complexity imposed by the new algorithm in two distinct domains, namely in terms of the execution time and the power consumption. Furthermore, we have analytically derived the upper and lower bounds of it complexity in both domains, which is on the order of $O(\sqrt{N})$ in both domains for the best-case scenario and on the order of $O(N\sqrt{N})$ and $O(N^2\sqrt{N})$ for the worst-case scenario in the execution time and power consumption, respectively. Explicitly, our new algorithm exhibits an optimal performance, despite its substantial execution time reduction compared to that of the NDQO algorithm, whilst imposing a similar power consumption.

REFERENCES

[1] B. Alawieh, Y. Zhang, C. Assi, and H. Mouftah, "Improving spatial reuse in multihop wireless networks—A survey," *IEEE Commun. Surveys Tuts.*, vol. 11, no. 3, pp. 71–91, Aug. 2009.
[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Commun. Mag.*, vol. 40, no. 8, pp. 102–114, Aug. 2002.
[3] X. Hong, K. Xu, and M. Gerla, "Scalable routing protocols for mobile ad hoc networks," *IEEE Netw.*, vol. 16, no. 4, pp. 11–21, Jul. 2002.
[4] S. Galli, A. Scaglione, and Z. Wang, "For the grid and through the grid: The role of power line communications in the smart grid," *Proc. IEEE*, vol. 99, no. 6, pp. 998–1027, Jun. 2011.
[5] D. Alanis, P. Botsinis, S. X. Ng, and L. Hanzo, "Quantum-assisted routing optimization for self-organizing networks," *IEEE Access*, vol. 2, pp. 614–632, 2014.
[6] X. Zhu, L. Shen, and T.-S. P. Yum, "Hausdorff clustering and minimum energy routing for wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 58, no. 2, pp. 990–997, Feb. 2009.
[7] M. Chen, V. C. M. Leung, S. Mao, Y. Xiao, and I. Chlamtac, "Hybrid geographic routing for flexible energy—Delay tradeoff," *IEEE Trans. Veh. Technol.*, vol. 58, no. 9, pp. 4976–4988, Nov. 2009.

[8] A. E. A. A. Abdulla, H. Nishiyama, J. Yang, N. Ansari, and N. Kato, "HYMN: A novel hybrid multi-hop routing algorithm to improve the longevity of WSNs," *IEEE Trans. Wireless Commun.*, vol. 11, no. 7, pp. 2531–2541, Jul. 2012.

[9] M. Al-Rabayah and R. Malaney, "A new scalable hybrid routing protocol for VANETs," *IEEE Trans. Veh. Technol.*, vol. 61, no. 6, pp. 2625–2635, Jul. 2012.

[10] F. Hoffmann, D. Medina, and A. Wolisz, "Joint routing and scheduling in mobile aeronautical ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 62, no. 6, pp. 2700–2712, Jul. 2013.

[11] H. Li, L. Lai, and H. V. Poor, "Multicast routing for decentralized control of cyber physical systems with an application in smart grid," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 6, pp. 1097–1107, Jul. 2012.

[12] G. A. Shah, V. C. Gungor, and O. B. Akan, "A cross-layer QoS-aware communication framework in cognitive radio sensor networks for smart grid applications," *IEEE Trans. Ind. Informat.*, vol. 9, no. 3, pp. 1477–1485, Aug. 2013.

[13] S. Canale, A. Di Giorgio, A. Lanna, A. Mercurio, M. Panfili, and A. Pietrabissa, "Optimal planning and routing in medium voltage PowerLine communications networks," *IEEE Trans. Smart Grid*, vol. 4, no. 2, pp. 711–719, Jun. 2013.

[14] H. Yetgin, K. T. K. Cheung, M. El-Hajjar, and L. Hanzo, "Cross-layer network lifetime optimisation considering transmit and signal processing power in wireless sensor networks," *IET Wireless Sensor Syst.*, vol. 4, no. 4, pp. 176–182, Dec. 2014.

[15] H. Yetgin, K. T. K. Cheung, M. El-Hajjar, and L. Hanzo, "Cross-layer network lifetime maximization in interference-limited WSNs," *IEEE Trans. Veh. Technol.*, vol. 64, no. 8, pp. 3795–3803, Aug. 2015.

[16] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing images using the Hausdorff distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 9, pp. 850–863, Sep. 1993.

[17] E. Sakhaee and A. Jamalipour, "The global in-flight Internet," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 9, pp. 1748–1757, Sep. 2006.

[18] L. Davis, *Handbook of Genetic Algorithms*. New York, NY, USA: Van Nostrand Reinhold, 1991.

[19] K. Deb, "Multi-objective optimization," in *Search Methodologies*, E. K. Burke and G. Kendall, Eds. New York, NY, USA: Springer-Verlag, 2005, pp. 273–316. [Online]. Available: http://dx.doi.org/10.1007/0-387-28356-0_10

[20] H. Yetgin, K. T. K. Cheung, and L. Hanzo, "Multi-objective routing optimization using evolutionary algorithms," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2012, pp. 3030–3034.

[21] M. Camelo, C. Omaña, and H. Castro, "QoS routing algorithm based on multi-objective optimization for wireless mesh networks," in *Proc. IEEE Latin-Amer. Conf. Commun. (LATINCOM)*, Sep. 2010, pp. 1–6.

[22] F. V. C. Martins, E. G. Carrano, E. F. Wanner, R. H. C. Takahashi, and G. R. Mateus, "A hybrid multiobjective evolutionary approach for improving the performance of wireless sensor networks," *IEEE Sensors J.*, vol. 11, no. 3, pp. 545–554, Mar. 2011.

[23] E. Masazade, R. Rajagopalan, P. K. Varshney, C. K. Mohan, G. K. Sendur, and M. Keskinoz, "A multiobjective optimization approach to obtain decision thresholds for distributed detection in wireless sensor networks," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 2, pp. 444–457, Apr. 2010.

[24] R. P. Feynman, "Simulating physics with computers," *Int. J. Theoretical Phys.*, vol. 21, no. 6, pp. 467–488, Jun. 1982.

[25] P. Benioff, "Quantum mechanical hamiltonian models of Turing machines," *J. Statist. Phys.*, vol. 29, no. 3, pp. 515–546, Nov. 1982. [Online]. Available: http://dx.doi.org/10.1007/BF01342185

[26] D. Deutsch, "Quantum theory, the Church–Turing principle and the universal quantum computer," *Proc. R. Soc. Lond. A, Math. Phys. Sci.*, vol. 400, no. 1818, pp. 97–117, Jul. 1985.

[27] D. Deutsch and R. Jozsa, "Rapid solution of problems by quantum computation," *Proc. R. Soc. Lond. A, Math. Phys. Sci.*, vol. 439, no. 1907, pp. 553–558, Dec. 1992.

[28] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. 28th Annu. ACM Symp. Theory Comput.*, 1996, pp. 212–219.

[29] M. Boyer, G. Brassard, P. Høyer, and A. Tapp. (1996). "Tight bounds on quantum searching." [Online]. Available: http://arxiv.org/abs/quant-ph/9605034

[30] C. Durr and P. Høyer. (1996). "A quantum algorithm for finding the minimum," [Online]. Available: http://arxiv.org/abs/quant-ph/9607014

[31] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Comput.*, vol. 26, no. 5, pp. 1484–1509, 1997.

[32] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, "Quantum algorithms revisited," *Proc. R. Soc. Lond. A, Math. Phys. Sci.*, vol. 454, no. 1969, pp. 339–354, Jan. 1998.

[33] G. Brassard, P. Høyer, and A. Tapp, "Quantum counting," in *Automata, Languages and Programming*. New York, NY, USA: Springer-Verlag, 1998, pp. 820–831.

[34] G. Brassard, P. Høyer, M. Mosca, and A. Tapp. (2000). "Quantum amplitude amplification and estimation." [Online]. Available: http://arxiv.org/abs/quant-ph/0005055

[35] A. Malossini, E. Blanzieri, and T. Calarco, "Quantum genetic optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 2, pp. 231–241, Apr. 2008.

[36] G. Brassard, F. Dupuis, S. Gambs, and A. Tapp. (2011). "An optimal quantum algorithm to approximate the mean and its application for approximating the median of a set of points over an arbitrary distance." [Online]. Available: http://arxiv.org/abs/1106.4267

[37] Z. Babar, S. X. Ng, and L. Hanzo, "Near-capacity code design for entanglement-assisted classical communication over quantum depolarizing channels," *IEEE Trans. Commun.*, vol. 61, no. 12, pp. 4801–4807, Dec. 2013.

[38] Z. Babar, S. X. Ng, and L. Hanzo, "EXIT-chart-aided near-capacity quantum turbo code design," *IEEE Trans. Veh. Technol.*, vol. 64, no. 3, pp. 866–875, Mar. 2014.

[39] Z. Babar, P. Botsinis, D. Alanis, S. X. Ng, and L. Hanzo, "The road from classical to quantum codes: A hashing bound approaching design procedure," *IEEE Access*, vol. 3, pp. 146–176, 2015.

[40] S. Imre and F. Balazs, *Quantum Computing and Communications: An Engineering Approach*. New York, NY, USA: Wiley, 2005.

[41] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge, U.K.: Cambridge Univ. Press, 2010.

[42] C. Zalka, "Grover's quantum searching algorithm is optimal," *Phys. Rev. A*, vol. 60, no. 4, pp. 2746–2751, Oct. 1999.

[43] G. Syswerda, "A study of reproduction in generational and steady state genetic algorithms," *Found. Genet. Algorithms*, vol. 2, pp. 94–101, 1991.

[44] J. Chiaverini *et al.*, "Implementation of the semiclassical quantum Fourier transform in a scalable system," *Science*, vol. 308, no. 5724, pp. 997–1000, May 2005.

[45] P. Botsinis, S. X. Ng, and L. Hanzo, "Quantum search algorithms, quantum wireless, and a low-complexity maximum likelihood iterative quantum multi-user detector design," *IEEE Access*, vol. 1, pp. 94–122, 2013.

[46] P. Botsinis, S. X. Ng, and L. Hanzo, "Fixed-complexity quantum-assisted multi-user detection for CDMA and SDMA," *IEEE Trans. Commun.*, vol. 62, no. 3, pp. 990–1000, Mar. 2014.

[47] P. Botsinis, D. Alanis, S. X. Ng, and L. Hanzo, "Low-complexity soft-output quantum-assisted multiuser detection for direct-sequence spreading and slow subcarrier-hopping aided SDMA-OFDM systems," *IEEE Access*, vol. 2, pp. 451–472, 2014.

[48] P. Botsinis, D. Alanis, Z. Babar, S. X. Ng, and L. Hanzo, "Iterative quantum-assisted multi-user detection for multi-carrier interleave division multiple access systems," *IEEE Trans. Commun.*, to be published.

[49] P. Botsinis, D. Alanis, Z. Babar, S. X. Ng, and L. Hanzo, "Noncoherent quantum multiple symbol differential detection for wireless systems," *IEEE Access*, vol. 3, pp. 569–598, 2015.

[50] Y. Han, D. M. Ancajas, K. Chakraborty, and S. Roy, "Exploring high-throughput computing paradigm for global routing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 1, pp. 155–167, Jan. 2014.

[51] S. Mu, X. Zhang, N. Zhang, J. Lu, Y. S. Deng, and S. Zhang, "IP routing processing with graphic processors," in *Proc. Design, Autom. Test Eur. Conf. Exhibit. (DATE)*, Mar. 2010, pp. 93–98.

[52] J. Zhao, X. Zhang, X. Wang, Y. Deng, and X. Fu, "Exploiting graphics processors for high-performance IP lookup in software routers," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 301–305.

[53] R. Mangharam and A. A. Saba, "Anytime algorithms for GPU architectures," in *Proc. IEEE 32nd Real-Time Syst. Symp. (RTSS)*, Nov. 2011, pp. 47–56.

[54] A. Uchida, Y. Ito, and K. Nakano, "An efficient GPU implementation of ant colony optimization for the traveling salesman problem," in *Proc. 3rd Int. Conf. Netw. Comput. (ICNC)*, Dec. 2012, pp. 94–102.

[55] U. Cekmez, M. Ozsiginan, and O. K. Sahingoz, "Adapting the GA approach to solve traveling salesman problems on CUDA architecture," in *Proc. IEEE 14th Int. Symp. Comput. Intell. Informat. (CINTI)*, Nov. 2013, pp. 423–428.
[56] T. Mohsenin, D. N. Truong, and B. M. Baas, "A low-complexity message-passing algorithm for reduced routing congestion in LDPC decoders," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 5, pp. 1048–1061, May 2010.
[57] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, "GPU computing," *Proc. IEEE*, vol. 96, no. 5, pp. 879–899, May 2008.
[58] J. D. Owens *et al.*, "A survey of general-purpose computation on graphics hardware," *Comput. Graph. Forum*, vol. 26, no. 1, pp. 80–113, Mar. 2007.
[59] L. L. Hanzo, S. X. Ng, W. Webb, and T. Keller, *Quadrature Amplitude Modulation: From Basics to Adaptive Trellis-Coded, Turbo-Equalised and Space-Time Coded OFDM, CDMA and MC-CDMA Systems*. New York, NY, USA: Wiley, 2004.
[60] M. Salem *et al.*, "An overview of radio resource management in relay-enhanced OFDMA-based networks," *IEEE Commun. Surveys Tuts.*, vol. 12, no. 3, pp. 422–438, Aug. 2010.
[61] S. Imre and L. Gyongyosi, *Advanced Quantum Communications: An Engineering Approach*. New York, NY, USA: Wiley, 2013.
[62] M. Saeedi and M. Pedram, "Linear-depth quantum circuits for *n*-qubit Toffoli gates with no ancilla," *Phys. Rev. A*, vol. 87, no. 6, p. 062318, Jun. 2013.

DIMITRIOS ALANIS (S'13) received the M.Eng. degree in electrical and computer engineering from the Aristotle University of Thessaloniki, in 2011, and the M.Sc. degree in wireless communications from the University of Southampton, in 2012, where he is currently pursuing the Ph.D. degree with the Southampton Wireless Group, School of Electronics and Computer Science.

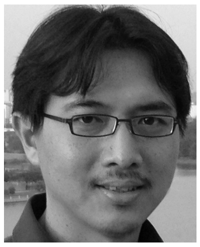
His research interests include quantum computation and quantum information theory, quantum search algorithms, cooperative communications, resource allocation for self-organizing networks, bioinspired optimization algorithms, and classical and quantum game theory.

PANAGIOTIS BOTSINIS (S'12) received the M.Eng. degree from the School of Electrical and Computer Engineering, National Technical University of Athens, Greece, in 2010, and the M.Sc. (Hons.) and Ph.D. degrees in wireless communications from the University of Southampton, U.K., in 2011 and 2015, respectively. Since 2010, he has been a member of the Technical Chamber of Greece. He is currently a Research Fellow with the Southampton Wireless Group, School of Electronics and Computer Science, University of Southampton.

His research interests include quantum-assisted communications, quantum computation, iterative detection, OFDM, multiple-input multiple-output, multiple access systems, coded modulation, channel coding, cooperative communications, and combinatorial optimization.

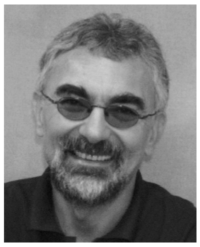
ZUNAIRA BABAR received the B.Eng. degree in electrical engineering from the National University of Science and Technology, Islamabad, Pakistan, in 2008, and the M.Sc. (Hons.) and Ph.D. degrees in wireless communications from the University of Southampton, U.K., in 2011 and 2015, respectively.

Her research interests include quantum error correction codes, channel coding, coded modulation, iterative detection, and cooperative communications.



SOON XIN NG (S'99–M'03–SM'08) received the B.Eng. (Hons.) degree in electronics engineering and the Ph.D. degree in wireless communications from the University of Southampton, Southampton, U.K., in 1999 and 2002, respectively. From 2003 to 2006, he was a Post-Doctoral Research Fellow working on collaborative European research projects known as SCOUT, NEWCOM, and PHOENIX. Since 2006, he has been a member of the Academic Staff with the School of Electronics and Computer Science, University of Southampton. He is currently an Associate Professor of Telecommunications with the University of Southampton. He is involved in the OPTIMIX and CONCERTO European projects and the IUATC and UC4G projects. He has authored over 180 papers and co-authored two John Wiley/IEEE Press books in his research field.

His research interests include adaptive coded modulation, coded modulation, channel coding, space-time coding, joint source and channel coding, iterative detection, OFDM, multiple-input multiple-output, cooperative communications, distributed coding, quantum error correction codes, and joint wireless-and-optical-fiber communications. He is a Chartered Engineer and fellow of the Higher Education Academy, U.K.



LAJOS HANZO (M'91–SM'92–F'04) received the degree in electronics in 1976, the Ph.D. degree in 1983, and the Doctor Honoris Causa degree from the Technical University of Budapest, in 2009. During his 38-year career in telecommunications, he has held various research and academic positions in Hungary, Germany, and U.K. Since 1986, he has been with the School of Electronics and Computer Science, University of Southampton, U.K., as the Chair in Telecommunications. He has successfully supervised about 100 Ph.D. students, co-authored 20 John Wiley/IEEE Press books in mobile radio communications totaling in excess of 10 000 pages, authored over 1500 research entries at the IEEE Xplore, acted as the TPC Chair and General Chair of the IEEE conferences, presented keynote lectures, and received a number of distinctions. He is currently directing 100 strong academic research teams, working on a range of research projects in the field of wireless multimedia communications sponsored by the industry, the Engineering and Physical Sciences Research Council, U.K., the European Research Council's Advanced Fellow Grant, and the Royal Society's Wolfson Research Merit Award. He is an enthusiastic supporter of industrial and academic liaison and offers a range of industrial courses.

He is a fellow of the Royal Academy of Engineering, the Institution of Engineering and Technology, and the European Association for Signal Processing. He is also a Governor of the IEEE VTS. From 2008 to 2012, he was the Editor-in-Chief of the *IEEE Press* and a Chaired Professor with Tsinghua University, Beijing. He has over 22 000 citations.