to add custom components into quantum workflows built with existing software libraries).

[0010] Finally, there are issues and challenges in quantum computing financial costs and VQA integration. As to costs, running computations on quantum hardware is expensive and time-consuming. Budgeting is a practical concern, exacerbated by the lack of automated methods to estimate the financial costs of running quantum computations. As to integration, the classical-quantum loop scales differently for both the size of the quantum processing unit (QPU) and its underlying technology. A generic VQA run requires exchanging information between quantum computers and classical computers. Without optimizing the interplay of the two computing units, the VQA run may suffer from network issues (like lost connectivity) and latency delays.

[0011] In the quest for quantum advantage, one needs better methods to create accessible and scalable VQA workflows. Thus, there is a need to provide methods, systems and kits for quantum-optimized cross-backend software development which address the drawbacks and challenges presently existing. In addition, there is a critical need for high-level software libraries and toolboxes to construct, optimize and test VQAs and other quantum algorithms such as Fault-tolerant Quantum Algorithms (FTQAs) on NISQ devices. Furthermore, other desirable features will become apparent from the subsequent detailed description and the appended claims, taken in conjunction with the accompanying drawings and this background of the disclosure.

SUMMARY

[0012] According to at least one aspect of the present embodiments, a method for building and executing algorithms in the Quantum Approximate Optimization

Algorithm (QAOA) family is provided. The method includes receiving an input comprising a user-defined problem and a specification of settings for the QAOA,

and converting the received input into a sequence of operations defining an abstract workflow. The method also includes setting-up and running a classical-quantum loop, the classical-quantum loop comprising: creating and executing the explicit representation of the quantum circuit based on the abstract QAOA representation, optimizing parameterized variables of the quantum circuit, and interrupting the loop once a satisfying training condition has been met. The method also includes returning an output, wherein the output comprises an optimization result and associated metadata.

[0013] According to another aspect of the present embodiments, a system for building and executing algorithms in the QAOA family is provided. The system includes an input module and a workflow module. The input module obtains an input model, where the input model includes a standardized version of a user-defined optimization problem and a specification of settings for the QAOA. The workflow module takes as input the input model and is configured to convert the input into a sequence of operations defining an abstract workflow that is independent of the target device upon which the computation is to be executed, and set up and run a classical-quantum loop. The classical-quantum loop is configured to create and execute the explicit representation of the quantum circuit based on the abstract QAOA representation, optimize parameterized variables of the quantum circuit, interrupt the loop once a satisfying training condition has been met. The workflow module is further configured to return an output comprising an optimization result and associated metadata.

[0014] According to another aspect of the present embodiments, a set of tools designed to improve the usability and performance of building quantum optimization algorithms is provided. The tools include a cost management tool configured to estimate costs associated with a given quantum computation for a defined combination of parameterization, optimizer, and decided backend, wherein costs comprise financial

costs, execution time and other metrics associated with running the given quantum computation, and an AutoVQA system configured to take the user-defined problem associated

with the given quantum computation and evaluates the optimal choices of parameters for a Variational Quantum Algorithm (VQA) statement of the user-defined problem. The tools also include a qubit routing tool to compile the Variational Quantum Algorithm representation in a backend-explicit optimized quantum circuit.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] The accompanying figures, where like reference numerals refer to identical or functionally similar elements throughout the separate views and which together with the detailed description below are incorporated in and form part of the specification, serve to illustrate various embodiments and to explain various principles and advantages in accordance with present embodiments.

[0016] FIG. 1 depicts a schematic illustration of functional elements and the application programming interface (API) structure for a system for constructing and executing Quantum Approximate Optimization Algorithm (QAOA) workflows (referred hereinafter as "OpenQAOA") in accordance with present embodiments.

[0017] FIG. 2 depicts an illustration of an anatomy of a workflow result for the Results Object of the OpenQAOA Core in accordance with the present embodiments.

[0018] FIG. 3 depicts a block diagram of operation of the Resource Estimator and Profiling Module enabler of the OpenQAOA in accordance with present embodiments.

[0019] FIG. 4 depicts a block diagram of operation of the Benchmarking Module enabler of the OpenQAOA in accordance with present embodiments.

[0020] FIG. 5 depicts an illustrative representation of a generic OpenQAOA Workflow in accordance with the present embodiments.

CLAIMS

What is claimed is:

1. A method for building and executing algorithms in the Quantum Approximate Optimization Algorithm (QAOA) family, the method comprising:

receiving an input comprising a user-defined problem and a specification of settings for the QAOA;

converting the received input into a sequence of operations defining an abstract workflow;

setting-up and running a classical-quantum loop, the classical-quantum loop comprising:

- creating and executing the explicit representation of the quantum circuit based on the abstract QAOA representation,

- optimizing parameterized variables of the quantum circuit,

- interrupting the loop once a satisfying training condition has been met; and

returning an output, wherein the output comprises an optimization result and associated metadata.

2. A set of tools designed to improve the usability and performance of building quantum optimization algorithms, the tools comprising:

a cost management tool configured to estimate costs associated with a given quantum computation for a defined combination of parameterization, optimizer, and decided backend, wherein costs comprise financial costs, execution time and other metrics associated with running the given quantum computation;

an AutoVQA, system configured to take the user-defined problem associated with the given quantum computation and evaluate the optimal choices of parameters for a Variational Quantum Algorithm (VQA) statement of the user-defined problem; and

a qubit routing tool to compile the Variational Quantum Algorithm representation in a backend-explicit optimized quantum circuit.

3. A system for building and executing algorithms in the QAOA family, the system comprising:

an input module for obtaining an input model, where the input model includes a standardized version of a user-defined optimization problem and a specification of settings for the QAOA; and

a workflow module that takes as input the input model and is configured to:

convert the input into a sequence of operations defining an abstract workflow that is independent of the target device upon which the computation is to be executed;

set up and run a classical-quantum loop configured to:

- create and execute the explicit representation of the quantum circuit based on the abstract QAOA representation,

- optimize parameterized variables of the quantum circuit, and

- interrupt the loop once a satisfying training condition has been met;

and

return an output, the output comprising an optimization result and associated metadata.

4. The system of claim 3 wherein the input module comprises a bank of common optimization problems, each of which can be called according to its standard name to reduce the user input burden.

5. The system of claim 3 wherein the input module supports the specification of a plurality of QAOA settings to subsequently allow a corresponding plurality of workflows to be created.

6. The system of claim 3 wherein the input module supports the specification of a device on which to run the computation (a ‘backend’), along with any relevant properties or characteristics of the backend.

7. The system of claim 6 wherein the workflow module comprises interfaces to deploy the quantum circuits to backend devices including cloud or on-premises quantum computers, and remote or local simulators of quantum computations.

8. The system of claim 7 wherein the system further comprises a benchmarking module containing a bank of pre-defined QAOA computations, each of which may be called by a unique reference, for the purposes of benchmarking the performance of the available backends.

9. The system of claim 6 wherein the workflow module comprises interfaces to additional modules for the purposes of qubit selection, mapping and routing.

10. The system of claim 3 wherein the workflow module comprises interfaces to additional external modules for the purposes of error mitigation.

11. The system of claim 3 wherein the input module supports the specification of, and the workflow module supports the use of, dynamic quantum circuit features such as mid-circuit measurements and feed-forward operations.