US Patent & Trademark Office
Patent Public Search | Text View

United States Patent Application Publication	20250258762
Kind Code	A1
Publication Date	August 14, 2025
Inventor(s)	NAVEH; Amir et al.

SETTING A QUANTUM STATE ON A QUBIT

Abstract

An apparatus, method, and system comprising: a quantum computer that is configured to execute a parametric quantum circuit a plurality of times, the parametric quantum circuit comprising: a qubit with an unknown quantum state at an initial cycle of the parametric quantum circuit, an ansatz parametric circuit for receiving the unknown quantum state and outputting a processed quantum state, and a detection sub-circuit for indicating a distance measure of the processed quantum state from a target state, wherein said quantum computer implements a Variational Quantum Algorithm (VQA) scheme to iteratively set parameter values of the ansatz parametric circuit until obtaining a parameter value that causes the parametric quantum circuit to output an approximation of zero; and an output module for outputting the ansatz parametric circuit.

Inventors: **NAVEH; Amir (Haifa, IL), UR; Shmuel (Shorashim, IL), MINERBI; Nir (Haifa, IL), KIRZNER; Ofek (Haifa, IL), GAZIT; Lior (Tel-Aviv, IL)**

Applicant: **Classiq Technologies LTD.** (Tel Aviv, IL)

Family ID: **80934030**

Appl. No.: **18/625689**

Filed: **April 03, 2024**

Related U.S. Application Data

parent US continuation 17354413 20210622 parent-grant-document US 11294797 child US 18625689
parent US continuation-in-part 17687087 20220304 parent-grant-document US 11960384 child US 18625689

Publication Classification

Int. Cl.: **G06F11/3698** (20250101); **G06F11/3668** (20250101); **G06N10/00** (20220101)

U.S. Cl.:

CPC **G06F11/3698** (20250101); **G06F11/3696** (20130101); **G06N10/00** (20190101);

Background/Summary

CROSS-REFERENCE TO RELATED APPLICATIONS [0001] This application is a Continuation In Part of, and claims the benefit of U.S. patent application Ser. No. 17/687,087, filed Mar. 4, 2022, titled “DEBUGGER FOR QUANTUM COMPUTERS”, which is a Continuation of U.S. patent application Ser. No. 17/354,413, now U.S. Pat. No. 11,294,797, filed Jun. 22, 2021, titled “DEBUGGER FOR QUANTUM COMPUTERS”, each of which are hereby incorporated by reference in their entirety without giving rise to disavowment.

TECHNICAL FIELD

[0002] The present disclosure relates to a quantum set operation in general, and to setting a target quantum state on a qubit with an unknown state mid-program, in particular.

BACKGROUND

[0003] Quantum computing is a computational paradigm that is fundamentally different from classical computing. In contrast to classical computing, which utilizes bits, quantum computing utilizes quantum bits (qubits). The qubits have unique features, as each qubit can be in superposition, several qubits can be entangled, and all operations on qubits besides measurement (referred to as quantum gates) must be reversible.

[0004] In quantum software, programming is done at the gate-level or near gate level. Quantum programming software and frameworks, such as Qiskit, CIRQ, Q #, FOREST, braket, silq, all allow user friendly interfaces for gate level programming. Additionally, quite a few building blocks and algorithms exist that perform certain pre-defined functionalities.

BRIEF SUMMARY

[0005] One exemplary embodiment of the disclosed subject matter is a system comprising: a quantum computer that is configured to execute a parametric quantum circuit a plurality of times, wherein the parametric quantum circuit comprises: at least one qubit with an unknown quantum state at an initial cycle of the parametric quantum circuit, an ansatz parametric circuit, the ansatz parametric circuit is configured to receive the unknown quantum state from the at least one qubit, whereby the ansatz parametric circuit is configured to output a processed quantum state, and a detection sub-circuit, the detection sub-circuit is configured to indicate a distance measure of the processed quantum state from a target state, wherein said quantum computer is configured to implement a Variational Quantum Algorithm (VQA) scheme to iteratively set parameter values of the ansatz parametric circuit until obtaining a parameter value that causes the parametric quantum circuit to output an approximation of zero as the distance measure; and an output module, said output module is configured to provide output data that corresponds to the ansatz parametric circuit.

[0006] Optionally, the system further comprises an original quantum circuit, the original quantum circuit comprising a qubit with the unknown quantum state, wherein a representation of the original quantum circuit is configured to be modified based on the output data.

[0007] Optionally, said modifying the representation comprises applying the ansatz parametric circuit to the qubit with the unknown quantum state during an intermediate cycle of an execution of the original quantum circuit.

[0008] Optionally, the target state is a predetermined quantum state.

[0009] Optionally, the target state is obtained from a user.

[0010] Optionally, the output data indicates the parameter value.

[0011] Optionally, the output data comprises a representation of the ansatz parametric circuit.
[0012] Optionally, the output data comprises a quantum circuit that reconstructs the ansatz parametric circuit according to the parameter value.
[0013] Another exemplary embodiment of the disclosed subject matter is a method comprising: obtaining a target state; obtaining a representation of an original quantum circuit, the original quantum circuit comprising a qubit with an unknown quantum state; generating a parametric quantum circuit, said generating comprises: setting the unknown quantum state on at least one qubit at an initial cycle of the parametric quantum circuit, generating an ansatz parametric circuit to receive the unknown quantum state from the at least one qubit, and to output a processed quantum state, generating a detection sub-circuit to indicate a distance measure of the processed quantum state from the target state, executing the parametric quantum circuit a plurality of times on a quantum computer, based on said executing, iteratively adjusting parameter values of the ansatz parametric circuit until obtaining a parameter value that, when executed by said executing, causes the parametric quantum circuit to output an approximation of zero as the distance measure, wherein said iteratively adjusting is part of a VQA scheme, and upon determining that the distance measure corresponds to the approximation of zero, returning output data corresponding to the ansatz parametric circuit; and adjusting the representation of the original quantum circuit based on the output data.
[0014] Optionally, said adjusting the representation comprises applying the ansatz parametric circuit to the qubit with the unknown quantum state during an intermediate cycle of an execution of the original quantum circuit.
[0015] Yet another exemplary embodiment of the disclosed subject matter is an apparatus comprising a processor and coupled memory, said processor being adapted to perform: obtaining a target state; obtaining a representation of an original quantum circuit, the original quantum circuit comprising a qubit with an unknown quantum state; generating a parametric quantum circuit, said generating comprises: setting the unknown quantum state on at least one qubit at an initial cycle of the parametric quantum circuit, generating an ansatz parametric circuit to receive the unknown quantum state from the at least one qubit, and to output a processed quantum state, generating a detection sub-circuit to indicate a distance measure of the processed quantum state from the target state, executing the parametric quantum circuit a plurality of times on a quantum computer, based on said executing, iteratively adjusting parameter values of the ansatz parametric circuit until obtaining a parameter value that, when executed by said executing, causes the parametric quantum circuit to output an approximation of zero as the distance measure, wherein said iteratively adjusting is part of a VQA scheme, and upon determining that the distance measure corresponds to the approximation of zero, returning output data corresponding to the ansatz parametric circuit; and adjusting the representation of the original quantum circuit based on the output data.
[0016] Yet another exemplary embodiment of the disclosed subject matter is a computer program product comprising a non-transitory computer readable medium retaining program instructions, which program instructions, when read by a processor, cause the processor to perform: obtaining a target state; obtaining a representation of an original quantum circuit, the original quantum circuit comprising a qubit with an unknown quantum state; generating a parametric quantum circuit, said generating comprises: setting the unknown quantum state on at least one qubit at an initial cycle of the parametric quantum circuit, generating an ansatz parametric circuit to receive the unknown quantum state from the at least one qubit, and to output a processed quantum state, generating a detection sub-circuit to indicate a distance measure of the processed quantum state from the target state, executing the parametric quantum circuit a plurality of times on a quantum computer, based on said executing, iteratively adjusting parameter values of the ansatz parametric circuit until obtaining a parameter value that, when executed by said executing, causes the parametric quantum circuit to output an approximation of zero as the distance measure, wherein said iteratively adjusting is part of a VQA scheme, and upon determining that the distance measure corresponds to

the approximation of zero, returning output data corresponding to the ansatz parametric circuit; and adjusting the representation of the original quantum circuit based on the output data.

Description

THE BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWINGS

[0017] The present disclosed subject matter will be understood and appreciated more fully from the following detailed description taken in conjunction with the drawings in which corresponding or like numerals or characters indicate corresponding or like components. Unless indicated otherwise, the drawings provide exemplary embodiments or aspects of the disclosure and do not limit the scope of the disclosure. In the drawings:

[0018] FIG. **1**A-**1**F show flowchart diagrams of methods, in accordance with some exemplary embodiments of the disclosed subject matter;

[0019] FIG. **2** shows a block diagram of an apparatus, in accordance with some exemplary embodiments of the disclosed subject matter;

[0020] FIGS. **3**A-**3**B show flowchart diagrams of methods, in accordance with some exemplary embodiments of the disclosed subject matter;

[0021] FIGS. **4**A-**4**C show illustration of quantum programs, in accordance with some exemplary embodiments of the disclosed subject matter;

[0022] FIG. **5** shows a flowchart diagram of a method, in accordance with some exemplary embodiments of the disclosed subject matter; and

[0023] FIG. **6** shows a flowchart diagram of a method, in accordance with some exemplary embodiments of the disclosed subject matter.

[0024] FIG. **7** shows a schematic block diagram, in accordance with some exemplary embodiments of the disclosed subject matter;

[0025] FIG. **8** shows an exemplary flowchart diagram of a method, in accordance with some exemplary embodiments of the disclosed subject matter;

[0026] FIG. **9** shows an exemplary block diagram of an apparatus, in accordance with some exemplary embodiments of the disclosed subject matter;

[0027] FIG. **10** shows an exemplary schematic block diagram, in accordance with some exemplary embodiments of the disclosed subject matter; and

[0028] FIG. **11** shows an exemplary flowchart diagram of a method, in accordance with some exemplary embodiments of the disclosed subject matter.

DETAILED DESCRIPTION

[0029] One technical problem dealt with by the disclosed subject matter is to provide for a quantum debugger. A quantum debugger may be a tool used to test and debug quantum programs. In some exemplary embodiments, the quantum debugger may be enabled to run a target quantum program and permit a user to track its operation, monitor its state or partial state, values of different variables or entities, or the like. In some exemplary embodiments, the quantum debugger may be designed to execute the quantum program on a quantum computer. Specifically, and as opposed to debuggers associated with simulators, the quantum debugger may utilize a quantum execution platform (e.g., the quantum computer) to execute, at least partially, the quantum program being examined. In some exemplary embodiments, the quantum debugger may enable to run or halt the target quantum program at specific points, display the contents of qubits, modify value of qubits, or the like. The quantum debugger may be utilized by a developer or other user to identify mistakes in the quantum program, such as by exploring the state of the program, in executing other options to enable a debugging process.

[0030] However, in order to look at the value of a variable in quantum programs, measurement of the value is performed, such as via sampling of a probabilistic result. When sampling is performed,

the distribution may collapse, which may affect the execution of successive cycles. In some exemplary embodiments, execution of the successive cycles after the distribution collapsed, provides a different result than that of the quantum program. In some cases, stopping a quantum program in the middle affects the quantum state, and prevents continuing execution therefrom, as may be desired as part of a debugging process.

[0031] It is further noted, that assertion-based debugging may be a naïve an inefficient debugging manner. Assertions may be added to the quantum program to understand, at different cycles, if desires properties are held. However, such debugging paradigm requires users to manually manipulate the quantum program and provides a limited set of abilities.

[0032] It may be desired to provide a quantum debugger that is capable of versatile abilities, including but not limited to stopping the program at desired locations or upon meeting desired conditions; inspecting a value of a variable mid-program; continuing execution of the program after value inspection; setting values to variables mid-program and continuing execution thereafter with the modified value; or the like.

[0033] In some cases, during debugging, the user may desire to investigate quantum-related properties, such as entanglement level between qubits, entanglement pattern, or the like. In some exemplary embodiments, quantum entanglement may be a physical phenomenon that occurs when a pair or group of particles are generated, interact, or share spatial proximity in a way such that the quantum state of each particle of the pair or group cannot be described independently of the state of the others.

[0034] Additionally or alternatively, all quantum operations should be reversible in quantum computing. However, setting of a value to a variable, mid-program, may not be a reversible operation. A technical solution for enabling setting a value of a variable without destroying desirable properties of the quantum program may be desired.

[0035] In general, the state of a qubit or the state of several qubits, within a circuit may be represented by a complex state vector of $\mathbf{2}^{n}$ complex numbers, where n represents the number of qubits whose state is inspected (which may include all qubits of the quantum program or portion thereof). In some exemplary embodiments, the quantum state may be a $2^{n}\times 2^{n}$ matrix, referred to as the density matrix. Measurements can, in theory given enough resources, fully reconstruct the density matrix, or state vector, including the phase, depending on how many resources are invested in the measurement. For the clarity of disclosure and without limiting the disclosed subject matter, the disclosed subject matter is described while focusing on a property that is derived from the density matrix-a distribution of different potential values. Such focus should not be construed as a limitation on the disclosed subject matter, which may relate to any property derived from the density matrix and to the content of the density matrix itself (e.g., the complex state vector of $\mathbf{2}^{n}$ complex numbers, the $2^{n}\times 2^{n}$ matrix itself, or the like). In some cases, any information derived from the density matrix, can be derived from measurements, and vice versa.

[0036] Another technical problem may relate to efficiently execute a quantum or non-deterministic program a plurality of times. In some cases, the same program may be re-executed to determine non-deterministic values, such as distribution of values, quantum states, or the like. Execution of the program may require substantial execution resources, and it may be desired to provide a more efficient manner of executing the same program over and over, while obtaining potentially different results.

[0037] As an example, a simulation software, implemented using classic software, may provide non-deterministic results. The simulation software may simulate traffic in a city. It may be desired to determine instead of or in addition to the specific value of variable in an execution, a distribution of values that the variable may receive. For example, it the traffic simulation, it may be desired to determine how many cars are used in the simulation. In different executions of the non-deterministic simulation, the number of cars may vary. In some exemplary embodiments, the user may inquire about a distribution of values of variable at a specific program location. In some

exemplary embodiments, the user may utilize a debugger and instruct the debugger to provide the distribution of values. Additionally or alternatively, the user may desire to review the final output of the program and review distribution thereof, with or without the use of a debugger. In all of these cases, in order to obtain the requested information, the classic software may need to be re-executed a plurality of times.

[0038] Similarly, when a value of a qubit is investigated, it may be required that the same quantum program be re-executed a plurality of times.

[0039] Yet another technical problem may relate to providing improved user experience (UX), when programming software that relate to quantum states and distribution of values. In some exemplary embodiments, the user may decide to set a distribution of values in a direct manner as part of the programming of the software. For example, the user may decide to program a classic software to receive values in accordance with a distribution. As another example, the user may desire to set a specific qubit to have a specific distribution. The specific distribution may be a-priori defined, may be based on the current value of the qubit being manipulated, may be based on the current value of another qubit being manipulated, may be based on current values of variables utilized by the program, or the like.

[0040] One technical solution is to execute the quantum program on the quantum computer a plurality of times, and during each time halting execution at a same target intermediate state. The target intermediate state may be defined based on a breakpoint, a conditional breakpoint, based on a previous state (e.g., “step” operation to reach a next state), or the like. In some exemplary embodiments, the target intermediate state may be defined as a specific cycle in the quantum program. In some exemplary embodiments, the intermediate state may be a state that precedes terminating states of the quantum program. The terminating state of the quantum program may be states that are the outcome of executing the last cycle (also referred to as “terminating cycle”) of quantum program. In some exemplary embodiments, the target intermediate state, therefore, may be any state that is defined in any cycle that is not the last cycle of the quantum program. In some exemplary embodiments, the intermediate state may be a state different than the state at which the quantum program is configured to finish execution.

[0041] In some exemplary embodiments, in each execution of the quantum program, a measurement of a target qubit (or a set of target qubits) may be performed. For example, a value of the qubit may be determined and used to determine a probability distribution of alternative values based on the probability that each value is measured. Additionally or alternatively, properties relating to the quantum state may be determined, such as but not limited to entanglement. Additionally or alternatively, quantum tomography may be performed. In some exemplary embodiments, quantum tomography may be the process by which a quantum state is reconstructed using measurements on an ensemble of identical quantum states. In some exemplary embodiments, the quantum tomography may be performed with a predetermined limited set of resources (e.g., number of executions, time, processing power, or the like). Additionally or alternatively, the quantum tomography may be performed until reaching a desired quality level (e.g., confidence level in measurement, estimated statistical deviation, or the like).

[0042] In some exemplary embodiments, based on the observed measurements obtained from actual execution of the quantum program by a quantum computer, when reaching the target intermediate state, the value of the target qubit (or set of target qubits) may be determined. In some exemplary embodiments, the number of executions performed may be determined based on a quality measurement, such as provided by the user, obtained from a predefined configurations or preferences, or the like.

[0043] In some exemplary embodiments, the value may be presented to the user. As an example, the user may be developer performing a debugging session and investigating the quantum program. The user may instruct a quantum debugger in accordance with the disclosed subject matter to display a value of a target qubit (or set of target qubits). In some exemplary embodiments, the

display may include any display of any property of the target qubit. For simplicity of disclosure, consider a display of a distribution of values of the set of qubits. In some cases, only non-negligible values may be displayed to the user, forming an abbreviated probability distribution, to provide a coherent display that can be comprehended by a human user, instead of a full probability distribution.

[0044] In some exemplary embodiments, the user may instruct the quantum debugger to continue execution after the target intermediate state. For example, the user may instruct the quantum debugger to execute a single cycle (e.g., “step” command). Additionally or alternatively, the user may instruct the quantum debugger to continue execution until reaching a next breakpoint (e.g., “run” command). Such commands may continue the execution of the quantum program in the same debugging session. However, as opposed to traditional debuggers, the quantum debugger may execute the quantum program a plurality of times until reaching a second target state (e.g., next cycle in case of a “step” command; a next cycle in which a breakpoint is defined in case of a “run” command; or the like) from an initial state and not from the intermediate state. In such a manner, although the execution appears to the user as being continued, it is actually restarting to ensure the quantum state was not modified (collapsed) to the measurements performed at the intermediate state.

[0045] In some exemplary embodiments, the user may instruct the quantum debugger to update a value of a qubit or set of qubits. For example, the user may execute a “set” command. In order for the operation to be reversible, instead of setting the value, a controlled propagation operation may be implemented, in which the value of the qubit is modified. A transformative quantum program that is configured to transform the value of the target qubit from its observed value (V1) to a target value (V2) may be synthesized. For example, the transformative quantum program may be configured to perform for the target qubit (Q) the following computation: $Q=Q-V1+V2$. In some cases, the transformation quantum program may be a program that is configured to transform a zero value (0) of the qubit to the modified value (V2) subtracted by the value (V1). The transformative quantum program may then be introduced to the quantum program in a manner causing the transformation of the value to be applied on an addition to the qubit value. In some exemplary embodiments, the transformative quantum program may be applied to the qubits, as a program, after the cycle in which the user asked for the controlled propagation command to be performed. In some exemplary embodiments, the quantum program may be updated so as to perform the transformative quantum program immediately after the target intermediate state. The modified quantum program that is created may be utilized for the remainder of the debugging session. Hence, when the user requests to continue execution (e.g., “step”, “run”), the modified quantum program may be executed and not the original quantum program. It is noted that such operation is different from traditional debuggers in which “set” operations update the memory space but do not alter the code section of the process executing the program.

[0046] In some exemplary embodiments, the user may set a conditional breakpoint for a debugging session. The conditional breakpoint may be set on a specific cycle or generally on all cycles. The conditional breakpoint may be associated with a condition on a value of a target qubit(s). In each cycle where the conditional breakpoint is applicable, the quantum debugger may determine the value of the target qubit, in accordance with the disclosed subject matter. The quantum debugger may evaluate whether the condition holds based on the value of the target qubit. If the condition holds the execution may be halted at the relevant cycle and a measurement may be obtained of the qubit(s) in the relevant cycle. In some exemplary embodiments, the user may utilize the GUI of the quantum debugger to investigate the properties of the program when the breakpoint is activated. For example, the user may request to view values of qubits. In some exemplary embodiments, the user may utilize the GUI to set values for qubits, perform “step” operations, or the like. Such operations may cause the quantum debugger to concatenate additional operations after the conditional breakpoint is activated (e.g., executing an additional cycle, synthesizing a

transformative quantum program to be executed after the breakpoint is activated, or the like). In some exemplary embodiments, the user may add new breakpoints, update or remove existing breakpoints, or the like.

[0047] In some exemplary embodiments, the quantum debugger may utilize previously determined value or property of a qubit in a specific state to determine value or property of the qubit in another state. In some exemplary embodiments, quantum program analysis may be employed.

[0048] As an example, based on a determination of quantum entanglement in cycle **150** between qubits Q1 and Q2, a determination of the quantum entanglements between the two qubits (Q1, Q2) in cycle **160** may be derived. For example, if in the intermediate cycles, both qubits were not manipulated in a manner affecting entanglement, it may be determined that the entanglement between them remains the same, even without executing the quantum program. As another example, a distribution of values of qubit Q1 on cycle **150** may be utilized to determine the distribution of values of qubit Q1 on cycle **160**. Consider if in the cycles between 150 and 160 there are only simple operations, the simple computations may be applied for each value in the distribution, and the target distribution in cycle **160** may be considered to be with the same probabilities and with the computed values.

[0049] In some exemplary embodiments, there may be a determination whether to utilize static analysis or dynamic analysis in order to determine values or properties of qubits. While the default may be to perform dynamic analysis, when the chance arises, and if it is considered as preserving or reducing resources required for the debugging process, static analysis may be utilize, in which quantum queries can be replied to, based on knowledge gleaned from previous queries.

[0050] In some exemplary embodiments, the static analysis may utilize information from succeeding cycles to determine values or properties. For example, using information from cycle **160**, a value or property regarding cycle **150** may be determined. This may utilize the reversibility property of quantum programs. It is further noted that in some cases, the user may instruct the quantum debugger to perform a “step back” operation, going back one cycle. As can be appreciated, this backward traversal is different from traversing an existing trace which was previously calculated and is merely presented, as may be the case in simulators. Instead, the disclosed subject matter, utilizes values from “future” cycles to determine values of “past′ cycles.

[0051] Another technical solution may be to utilize impact analysis on the program to determine potential portions thereof that can impact the investigated result. Instead of re-executing the entire program, the program may be modified without affecting the outcome. In some exemplary embodiments, lightcone analysis of a quantum program may be performed to identify potential portion of the quantum program that can be dropped. As another example, the analysis may be aware of quantum properties of different quantum gates. Some gates may be considered as “partial impacting gates”, such as SWAP gate, where the outcome of different outputs of the gate may be affected by only a portion of the qubits that are inputted thereto. In contrast, full impacting gates, such as CNOT gates, CZ gates, Toffoli gates, or the like, may be affected and potentially affect the value of each qubit inputted thereto. Based on the differentiation between partial impacting gates and full impacting gates, different impact analysis may be performed.

[0052] In some exemplary embodiments, in view of the impact analysis, portion of the program that do not affect the investigated variable may be removed to reduce required resources for execution. For example, the number of cycles in a quantum program may be reduced, the number of qubits utilized in a quantum program may be decreased, the amount of memory to be utilized by a classic program may be reduced, certain functions and methods may be omitted and execution thereof may be spared in a classic program, or the like. The modified program may be compiled and executed instead of the original program.

[0053] In some exemplary embodiments, preprocessing may be performed to create a modified program, P′, instead of the original program P. The modified program may require reduced amount of resources, compared to P, while being equivalent to P with respect to a partial execution result

that is of interest. In some exemplary embodiments, execution of P′ may be faster, utilize reduced amount of memory or other resources, or the like. In some exemplary embodiments, there are multiple optimization that can be performed to create modified program P′. In some exemplary embodiments, different optimizations may be different pre-processing overhead and may yield different execution improvements. An optimization may be denoted O.sub.i, the modified program obtained based on performing optimization O.sub.i may be denoted as P(O.sub.i). Additionally or alternatively, a plurality of optimizations may be utilized in conjunction. As an example, if both optimizations O.sub.i and O.sub.j are implemented the modified program obtained may be P(O.sub.i, O.sub.j). Preprocessing overhead of performing optimization O.sub.i may be denoted as C(O.sub.i). Execution cost of executing a program P may be denoted as C(P), hence C(P(O.sub.i)) may denote the execution cost of a single execution of the modified program obtained by implementing optimization O.sub.i. In some exemplary embodiments, the optimizations may be implemented in order to reduce the costs of obtaining the partial execution result that is of interest. In accordance with the disclosed subject matter, the partial execution result may require N executions. Hence, the entire cost of obtaining the partial execution result using the original program may be denoted as N.Math.C(P). Using optimization O.sub.i, the execution cost is C(O.sub.i)+N.Math.C(P(O.sub.i)). Hence, the optimization O.sub.i should be used only if C(O.sub.i)+N.Math.C(P(O.sub.i))<N.Math.C(P). In some cases, the preprocessing costs and the execution benefit (e.g.,
[00001]$\frac{C(P(O_i))}{C(P)}$)
may be unknown in advance. In some exemplary embodiments, prediction of the expected costs may be utilized, such as based on the problem and statistics. In some exemplary embodiments, the prediction may also take into account the execution platform, compiler information and gate usage statistics by the program, as in some cases different gates may yield different performance on different execution platforms. If an optimization can eradicate the use of a specific gate type that performs poorly on the existing execution platform, such optimization may be predicted to yield a relatively significant performance improvement. In some exemplary embodiments, a quality and cost evaluation may obtain the program and one or more potential optimizations, and provide an estimation as for the expected overhead cost as well as estimated reduction in execution costs in view of the optimization. Based on the number of iterations (N) to be performed, based on estimated or measured execution cost of the original program (C(P)), and based on the estimated overhead costs and reduction in execution costs for each potential optimization (or combination of optimizations), the optimization to be performed may be selected. In some cases, after the optimization is performed, the actual metrics may be measured and utilized to improve future predictions, selections, or the like. In some exemplary embodiments, if the estimated outcome of the optimization and the actual measured outcome differ, it may be re-evaluated whether to select a different optimization. Assuming optimization O.sub.i was selected, a different optimization (or combination of optimizations) O.sub.j may be selected and applied on the original program in case C(O.sub.j)+N.Math.C(P(O.sub.i))<N. C(P(O.sub.i)). In some cases, the optimization may be performed on the optimized program P(O.sub.i). Such optimization may be selected if C(O.sub.j)+N.Math.C(P(O.sub.i, O.sub.j))<N.Math.C(P(O.sub.i)). In some cases, the optimization may be performed on the original program instead of on the optimized program if C(P(O.sub.j)) <C(P(O.sub.i, O.sub.j)).

[0054] Yet another technical solution is to enable a programmer to utilize operations to propagate a quantum state to a qubit, a distribution value, or the like, in a controlled manner. In response to such an instruction, a transformation quantum program that is configured to modify the current distribution value to become the target distribution value may be generated and added to the quantum program. In such a case, the programmer may be provided with a relatively easy interface to provide complicated coding instructions.

[0055] Similarly, programmers of classic software, such as simulation software, may also make use

of such instructions. In some exemplary embodiments, the programmer may define, such as in a debugger or when coding the program itself, to set a distribution value to a variable. For example, the instruction may be x=Distribution D {(20%, 3), (50%, 5), (30%, 8)}. During execution of the program, the value of variable x may be sampled from distribution D. In each execution, the value of x may differ, but overall the values of x, when set, would have the desired distribution. In some cases, x may already have a certain distribution, defined explicitly (e.g., using another set distribution operation) or implicitly (e.g., as a result of non-deterministic operation of the code). The programmer or user may set or modify the distribution value of x over all executions, based on its original distribution. As an example, the operation may manipulate the value of x (e.g., increase x by 1, would change distribution D above to D′ {(20%, 4), (50%, 6), (30%, 9)}). As another example, the operation may manipulate only some of the values. As an example, if only the value of 3 is multiplied by ten, distribution D may be modified to D″ {(20%, 30), (50%, 5), (30%, 8)}. In some cases, the probabilities of each value may also be modified, such as setting a specific value to be a distribution. As an example, instead of the value 3, a distribution D.sub.2{(50%, 3), (50%, 5)} may be set. In such an example, distribution D may be modified to D″ {(10%, 3), (60%, 5), (30%, 8)}. In some exemplary embodiments, the value of x may be manipulated using values, singular or distribution, of other variables, such as y. referring to the example above, D.sub.2 may be the distributed value of y.

[0056] One technical effect of utilizing the disclosed subject matter is to provide a solution enabling debugging of quantum programs. Such solution provides for direct debugging of the program when executed on quantum computer, and does not rely on simulators. As quantum computers are utilized, more complex program can be executed, without requiring a traditional computer to simulate quantum computations. The solution may provide for “standard” debugging user interface (e.g., enabling debug commands such as “step”, “run”, “break”, “conditional break”, “set”, “step back”, etc.), in a non-standard computerized environment.

[0057] Another technical effect of utilizing the disclosed subject matter is providing for human-comprehensible output, by disregarding negligible results. As quantum computers may inherently involve some statistical error in values, displaying all values may be redundant and prevent a human user from being able to perform meaningful debug operation. The disclosed subject matter, enables the human user to comprehend the meaningful information while blocking non-meaningful information from view.

[0058] Yet another technical effect is the ability to compute values for qubits based on values of qubits in other cycles, which may be succeeding or preceding cycles. The analysis used can reduce a number of executions of the quantum computer required in order to obtain information that meets a desired quality level.

[0059] Yet another technical effect may be to reduce required resources to investigate a quantum program. In some exemplary embodiments, the modified quantum program that is generated may include a reduced amount of qubits, cycles, or the like. In some exemplary embodiments, the modified quantum program may be executed using reduced resources of the execution platform, in comparison with the original quantum program.

[0060] Yet another technical effect may be to reduce required resources to investigate a classic non-deterministic program that has distribution of values over several executions. The user may investigate the distribution of values, manipulate such distribution using “controlled propagation” operation regarding distributions, or the like. In some exemplary embodiments, such investigation may be performed by executing the program a plurality of times, and the disclosed subject matter may improve performance and reduce overall required resources, such as time, CPU, memory, or the like, by modifying the program and executing the modified program instead of the original program.

[0061] The disclosed subject matter may provide for one or more technical improvements over any pre-existing technique and any technique that has previously become routine or conventional in the

art. Additional technical problems, solutions, and effects may be apparent to a person of ordinary skill in the art in view of the present disclosure.

[0062] Referring now to FIGS. **1**A-IF showing flowchart diagrams of methods, in accordance with some exemplary embodiments of the disclosed subject matter. Reference is initially made to FIG. **1**A.

[0063] On Step **110**, a Quantum Program (QP) may be obtained. The quantum program may be a program to be debugged. In some exemplary embodiments, a quantum program may be constructed out of layers. Each layer may be a tensor product of a certain fixed set of gates. A quantum program may be a matrix, defined by a product of layers L.sub.1. L.sub.2 . . . . L.sub.d. The number of layers d may be referred to as the depth of the circuit. In some exemplary embodiments, a quantum program over n qubits may be a unitary operator U.sub.n. In some exemplary embodiments, U.sub.n may be a matrix of 2.sup.n. 2.sup.n. U.sub.n may be a unitary matrix, a matrix that is if multiplied by a vector with norm of 1, outputs a vector with norm of 1. In some cases, a unitary matrix may be a quantum program, in which unitary operations (gates) are applied on the qubits. Additionally or alternatively, valid quantum state, denoted as, may be a vector of size 2.sup.n with a norm of 1. The execution of the quantum program may be the multiplication of the state ψ by the unitary matrix U.sub.n, resulting in an output state. It is noted that the gates may comprise logical gates (e.g., AND, OR, XOR, or the like), qubit gates (e.g., U.sub.3(θ, φ, λ) gate, Hadamard gate (H), controlled not gate (CX, CNOT), controlled Z gate (CZ), toffoli gate, swap gate, phase shift gate (R.sub.φ), Pauli-X gate (X), Pauli-Y gate (Y), Pauli-Z gate (Z), or the like), or the like.

[0064] On Step **120**, the user may provide an instruction to the quantum debugger. In some exemplary embodiments, the user may utilize a Graphical User Interface (GUI) or another Human-Machine Interface (HMI) to provide instructions to the quantum debugger. In some exemplary embodiments, the user may instruct the debugger to add a non-conditional breakpoint at a target cycle (C). Such an instruction may cause the quantum debugger to stop execution when reaching cycle C. It is noted that many breakpoints may be set within the same debugging session. The first breakpoint that is reached and which is active (e.g., in case of a conditional breakpoint) may cause the execution to stop at the corresponding cycle.

[0065] On Step **130**, the user may instruct to view the value of a target qubit, denoted as Q. The instruction may be provided via the GUI of the quantum debugger. It is noted that in some cases, until such instruction is provided, the quantum program may not be executed. This may be the case even if the quantum debugger provides the user with a GUI indicating that the execution has started and was halted at cycle C in view of the breakpoint defined on Step **120**. In some exemplary embodiments, the execution of the program may be performed only when a debug statement, such as “print value”, “set”, or the like, are required to be performed.

[0066] Steps **140**-**170** may be performed to satisfy the user instruction of Step **130**. On Steps **140**-**160**, the quantum program may be executed on a quantum computer a plurality of times. In each execution, the quantum program may not be allowed to complete execution (e.g., reach a last cycle of the quantum program), but instead may be halted when reaching cycle C(Step **140**). The quantum state which exists when reaching cycle C may be referred to as an intermediate state as such state is intermediate between an initial state and a last state of the quantum program. The value of the qubit(s) Q may be measured in the intermediate state (Step **150**). The number of executions of the quantum program may be determined based on whether a desired quality measurement was reached (Step **160**). The quality measure may relate to statistical properties of the gathered information, such as the standard deviation of the measurements, estimated confidence in the result, or the like. In some cases, when performing more executions and gathering more information, it may be determined the amount of added information gained by such additional executions to determine if the target quality measurement was reached. Additionally or alternatively, instead of or in addition to tracking quality measurement, a time limit or another limit on utilized resources may be determined and upheld, such that that when the resources are

exhausted, no additional executions are performed. The gathered results may be aggregated and displayed to the user (Step **170**) as a response to the user's instruction of Step **130**.

[0067] In some exemplary embodiments, the debug command may instruct displaying information regarding a plurality of qubits. In some exemplary embodiments, the value may be a density matrix. Additionally or alternatively, the displayed information may be derived from the density matrix (or computed directly). As an example, the user may be interested in a distribution of values of the set of qubits. The size of the distribution may be exponential in the number of qubits. In some exemplary embodiments, If the user asks to see the distribution on a relatively small number of qubits (e.g., below a threshold), such a distribution may be measured by testing. If the number of qubits chosen is large (e.g., above the threshold), then displaying the entire distribution may be problematic as the information may be non-comprehensible to a human user. Values that are never seen (e.g., 0 instances), or values with low probability (e.g., having a probability below a threshold), may be omitted from the display.

[0068] In some exemplary embodiments, in order to determine the distribution, the operation may receive as input the quantum program, a location and a set of n qubits to be evaluated, a quality measure, a limit on resources, or the like. The created distribution may be a set of valuation evaluated n qubits where each valuation has a value in (0,1).sup.n and probability (e.g., a real number attached to the valuation) such that the sum of the real numbers of all the non-zero valuation is less than one. As an example, the distribution may be, for example, five valuations with total probability of 60%, and the rest (i.e., remaining 40%) are unknown. In some exemplary embodiments, the operation may fail to be done in a time limit or within a limitation on available resources. Additionally or alternatively, there may be a limit on the number of valuations shown in the distribution, such as defined by the user expecting to see not more than a predetermined number of values. In some cases, it may find more states than the valuation limit, in which case the operation may stop.

[0069] In some exemplary embodiments, if the number of qubits is small (e.g., below a threshold of 3 qubits, 4 qubits, 5 qubits, or the like), the quantum program may be executed a multiple time, for example 1024 times, to gather observed valuations and assign to each of which a respective probability based on the number of times it was observed.

[0070] In some exemplary embodiments, if the number of qubits is large (e.g., above the threshold), the potential number of valuations may be huge. For example, if there are 100 qubits, and the quantum program is simply applying gate H to each, the size of the state space may be 2.sup.100 valuations all having positive, equal, probability. Such a space may be hard to present to the user in an explicit manner. In some exemplary embodiments, the potential number of valuations may be huge, but only a small subset thereof has non-negligible probability. It is noted that zero probability may be considered negligible. However, due to statistical errors, valuations which theoretically should not be provided by the quantum program, may be observed in reality. Such valuations may be observed a relatively small number of times, which may be considered as negligible probability. In some exemplary embodiments, negligible and non-negligible values may be distinguished based on their respective probabilities being below or above a probability threshold, respectively (Step **172**). The output presented to the user may be limited to only display non-negligible values (Step **147**), thereby providing a human-comprehensible result. The probability threshold may be set by a user, determined automatically, or the like. In some exemplary embodiments, the probability threshold may be determined based on an estimated error rate; based on a number of potential valuations; based on a mean, median, average of the probabilities in the distribution; based on predetermined percentile of the probabilities in the distribution, such as 30% percentile, 40% percentile, 60% percentile, 80% percentile, or the like; or the like.

[0071] In some exemplary embodiments, the debugger may execute the quantum program multiple times to provide an output to the user. With the sampled results the debugger gets a set of

valuations each with the number of times it was found. For example, after running 100 times and getting value v.sub.1 50 times and other 50 values (V2 . . . . V51) each a single time. Based on such information, the disclosed subject matter may estimate the probability of value v.sub.1 to be about 50%. As for the probability of each other value-no information may be determined due to a small sample (e.g., below a threshold of 30 instances, for example). In order to investigate the additional valuations, the quantum program may be modified to prevent the value of v.sub.1, for which sufficient information is available to provide a probability with a confidence measurement that is above a minimal threshold. After such modification, the modified quantum program may be executed to investigate other valuations with significant lower probabilities. For example, consider determining after an initial examination that 50% of the time, the value is v.sub.1 and 30% of the times the value is v.sub.2. The modified quantum program may be modified so as to prevent the values of v.sub.1 and v.sub.2. Hence, the modified quantum program may provide the relative probability of the remaining 20% of the valuations. If value v.sub.3 is identified as appearing 25% of the time in the modified quantum program, the value v.sub.3 may be assigned the probability of 5% (20%, 25%=5%). Such modification enables the disclosed subject matter to identify the distribution using a reduced number of executions. The modified quantum program may be more efficient in finding valuations with lower probabilities. In some exemplary embodiments, the modified quantum program is utilized only to determine the value of the set of qubits being investigated. When additional examinations are performed, such as in response to another debug command from the user, the modified quantum program may not be utilized. Instead, the quantum program that was analyzed may be utilized. Put differently, the modified quantum program utilized to examine less common valuations is used only to provide the response to the debug command and is not continued to be used during the same debugging session.

[0072] In some exemplary embodiments, quantum tomography may be utilized to reconstruct a quantum state (or portion thereof) using measurements on an ensemble of identical quantum states. The source of these quantum states may be any device or system which prepares quantum states either consistently into quantum pure states or otherwise into general mixed states. In some exemplary embodiments, given a set of qubits, tomography may be utilized to interrogate and find their state. In some exemplary embodiments, with enough measurement, the state of the qubits can be determined was using measurement. In some exemplary embodiments, the user may specify what he wants to see. For example, the user may provide commands to the quantum debugger indicating desired properties for the tomography process. For example, the user may limit the time or resources utilized for measurement, target accuracy, specific information desired (e.g., information on axis X, axis Y, axis Z, combination thereof, or the like), minimal probability of valuation to be displayed, properties of the density matrix, entropy of measurements, entanglement between qubits or other quantum properties, or the like.

[0073] In some exemplary embodiments, tomography may be performed by adding gates after the quantum state, to assist in measuring the quantum state. As an example, if the accuracy of the measurement is between 6 and 7, while the actual number is 6.4, a gate adding 0.5 can be applied, and if the measurement thereafter is still **6**, then the value is between 6 and 6.5. Such process can be continued, such as using binary search, until reaching a desired level of accuracy.

[0074] In some exemplary embodiments, it may be checked whether the quantum state obtained is pure or not. Quantum states that cannot be written as a mixture of other states are called pure quantum states, while all other states are called mixed quantum states. A pure quantum state can be represented by a ray in a Hilbert space over the complex numbers.

[0075] Referring now to FIG. **1**B, showing an embodiment in which the number of executions is determined a-priori (Step **160**B). Steps **140**-**150** are performed a number of times that was determined on Step **160**B. It is noted that the determination may be based on predetermined time limit, resource limit, target quality measurement, or the like. Additionally or alternatively, the number of times to execute may be modified after executing some or all of the executions of Steps

140-**150**.
[0076] Referring now to FIG. **1**C, in which the multiple executions are represented by Step **140**C. After the quantum debugger provides the response to the user instruction that was provided on Step **130**, e.g., after Step **170**, the user may provide additional debug commands to the quantum debugger.
[0077] On Step **130**C, the user provides an instruction to propagate a target value to qubit(s) Q. The target value may be a modified probability distribution, a modified density matrix, or the like. In some exemplary embodiments, the "controlled propagation" instruction may be an instruction to set the distribution to a predetermined or pre-calculated distribution; to set the distribution taking existing results into account, such as by changing values in the distribution without affecting their probability or by changing values or probabilities of existing values in the distribution; to set the distribution taking into account other variable(s), such as other qubits and their density matrix or distribution; or the like.
[0078] In some exemplary embodiments, the propagating of a different distribution may enable the user to define a distribution in an explicit manner. For example, the input may be a set of pairs (p.sub.i, v.sub.i), where p.sub.i is the probability and v.sub.i is the valuation of the i-th entry in the distribution. Additionally or alternatively, the user may provide a quantum pure state, which may be represented by a vector. Additionally or alternatively, the state can be a mixed, non-pure, state, which may be represented using a matrix. Additionally or alternatively, the new value may be determined using the existing distribution. For example, given original distribution represented by pairs (p.sub.i, v.sub.i), where p.sub.i is the probability and v.sub.i is the valuation, the new distribution may be computed using the function f.sub.i such that the new distribution (p.sub.i, f.sub.i (v.sub.i)). In some cases, f.sub.i may be identical for all i values or may be different for different i values. Consider the following example of distribution: D={(20%, 5), (80%, 8)}. The new distribution may increase all values by 1, using the following functions: f.sub.1(*x*)=f.sub.2(*x*)=x+1. The new distribution may increase only the value of 5 by 1 using the following functions: f.sub.1 (*x*)=x+1; f.sub.2 (*x*)=x. The new distribution may split the value of 5 to 5 in 30% of the cases and to 6 in the remaining 70% using the following functions: f.sub.1(x)= {.sub.x 30%.sup.x+1 70% . Applying such function may result in the following distribution: {(14%, 6), (6%, 5), (80%, 8)}. In some exemplary embodiments, the operation may depend on the value of another variable, such as qubit Z.
[0079] On Step **180**C, the quantum debugger may synthesize a transformative quantum program, denoted TQP. The TOP may be a quantum program snippet that is configured to modify the zero value to a predetermined value. The predetermined value may be the difference between the target value and the existing value. Put differently, if the value determined on Step **170** is V1, and the target value is V2, the TOP may be configured to transform the zero value to V2-V1.
[0080] In some exemplary embodiments, the TOP may be computed in accordance with the following scheme. Assume we begin with a pure state (1, and we wish to create a quantum sub circuit (the TQP) that takes φ.sub.1 to φ.sub.2. U.sub.1 may be a unitary that takes 41 and transforms it into the zero state |00 . . . 0). U.sub.2 may be a unitary that takes the zero state |00 . . . 0) and transforms it into φ.sub.2.Math. Accordingly, given state φ.sub.1, state φ.sub.2 can be computed as follows: φ.sub.2=U.sub.2U.sub.1φ.sub.1. Put differently, the TOP can be based on U=U.sub.2U.sub.1. U may be decomposed to primitive gates, using known methods, to synthesize the desired circuit, TQP.
[0081] In some exemplary embodiments, U.sub.2 can be built as follows. The first column of the matrix U.sub.2 may be set to be 42. The matrix U.sub.2 may be completed into a unitary matrix. In some exemplary embodiments, this can be achieved by filling the next columns of U.sub.2 with random complex numbers to receive the matrix and perform ortho-normalization of the columns of the matrix, using the Gram-Schmidt method. After this process U.sub.2 is unitary, and is configured to receive the zero state and calculate state φ.sub.2.

[0082] In some exemplary embodiments, U.sub.1 can be built using a similar technique by creating the matrix that receives the zero state and calculates state φ.sub.1. U.sub.1 may be the inverse of the calculated matrix. Inversing the matrix may be performed by decomposing the matrix to basic gates and then synthesizing a circuit in which the gate order is reversed and each gate is inversed.
[0083] In some cases, transition between quantum states may be implemented without the use of any auxiliary qubits. Additionally or alternatively, in some cases, such as for example in mixed quantum states, the transition may require the usage of auxiliary qubits. In some exemplary embodiments, the TQP may utilize more qubits than the program itself utilizes. Additionally or alternatively, the TOP may require to allocate a new auxiliary qubit to be utilized for its operation.
[0084] On Step **182**C, the TOP may be inserted into the quantum program, thereby creating am modified quantum program. The TOP may be inserted so as to add to the target qubit(s) (Q) the difference value. In some exemplary embodiments, the modified quantum program may be configured to update the value of Q to be Q=Q+V2-V1. As Q, at the target cycle, C, is valued V1, Q is updated to be Q=V1+V2-V1=V2.
[0085] On Step **184**C, and similarly to Step **140**C, the modified quantum program may be executed a plurality of times to obtain execution results.
[0086] Referring now to FIG. **1**D, in which the user does not provide instruction to display the value of Q at cycle C, but merely update the value of Q. Still, Step **140**C may be performed to determine the initial value of Q.
[0087] It is noted that the modified quantum program may continue to be used for the duration of the debugging session. Hence, the effect of the modification of the quantum program, may appear to persist in the remainder of the debugging session. While the memory area is not updated by the quantum debugger, the user may perceive such modification as appearing to reflect a similar memory update.
[0088] FIG. **1**D exemplifies this in Steps **120**D, **122**D, **184**D.
[0089] After the steps associated with the controlled propagation command are performed (Steps **140**C, **180**C, **182**C), the user may proceed in the same debugging session and provide additional debug commands (**120**D, **122**D). In Step **120**D, a breakpoint is set in another cycle (C2), which succeeds the target cycle (C) (e.g., C2>C). In Step **122**D, the user instructs the quantum debugger to continue execution. Such commands are performed (Step **184**D) while executing the modified quantum program, and not the original quantum program of Step **110**, until reaching cycle C2, in which measurements may be taken to provide execution results to the user. It is noted that if further "controlled propagation" commands are performed, the quantum program may be further modified by introducing additional transformative quantum programs thereto.
[0090] Referring now to FIG. **1**E, in which the user may provide a debug command instructing to introduce a conditional breakpoint at a target cycle, C, with a condition, COND. If the condition relates to non-quantum elements, the conditional breakpoint may be evaluated using a single execution of the quantum program. Additionally or alternatively, if the condition relates to quantum elements, such as qubits, the evaluation of the condition may require multiple executions. The quantum program may be executed several times (Step **140**C), and the condition may be evaluated to determine if the condition holds (Step **170**E). Appropriate output may be presented to the user (Step **180**E).
[0091] As another example, the conditional breakpoint may be to present the value of qubit Q2 if the value of qubit Q1>q. If Q2 has more bits than Q1 or more complicated distribution, the number of executions required to present the value of Q1 may be different. Assume Q1 is one qubit, and Q2 comprises 10 qubits. First, the quantum program is executed to see if the condition of Q1>q is held. During such executions, the value of Q1 is measured. Potentially, also information about Q2 is obtained during such executions. However, if the condition is held, additional executions may be required in order to collect sufficient measurements regarding Q2, such as additional 1000 executions.

[0092] Referring now to FIG. **1**F. The value of qubit Q at cycle C may be determined (Step **140**F) based on multiple executions of the quantum program (Step **140**C). Using the information obtained at cycle C, it may be determined if the value of Q at a different cycle (C2) can be determined using static analysis. If so, static analysis is performed (Step **165**F) otherwise, dynamic analysis, in which the quantum program is executed, is utilized (Step **160**F).

[0093] In some exemplary embodiments, based on information that was gathered at cycle **150**, a query regarding cycle **160** may be determined without executing the quantum program. For example, assume Q1 and Q2 are entangled on cycle **150**. If between cycles **150** and **160** no operation that affect entanglement is performed regarding such qubits, it may be determined that Q1 and Q2 remain entangled. As another example, if the distribution of Q1 is calculated on cycle **150**, and if only simple operations (e.g., non-quantum operations, such as logical gates) are applied on Q1 between cycles **150** and **160**, the distribution of Q1 on cycle **160** can be computed without executing the quantum program. For example, assuming the distribution of Q on cycle **150** is the list of pairs (p.sub.i, v.sub.i), where p.sub.i is the probability and v.sub.i is the valuation, and assuming the non-quantum function applied on Q between cycles **150** and **160** is f ( ) then the distribution of Q in cycle **160** may be computed statically as the list of pairs (p.sub.i, f(v.sub.i)).

[0094] In some exemplary embodiments, a graph of entanglement properties may be generated in which nodes represent qubits and edges represent entanglement between the qubits. In some exemplary embodiments, weights of the edges may represent properties of the entanglement relationship between the two qubits, such as phase, number of shmidt coefficients, Von-neumann entropy, or the like.

[0095] An entanglement graph computed for cycle **150** may remain unmodified until cycle **160** if the quantum program does not apply any gate that affects entanglement. Additionally, or alternatively, even if such gates are included in the quantum program between cycles **150** and **160**, they may only affect a subset of qubits. The sub-graph consisting of the remaining qubits that are unaffected by such gates (e.g., the gates are not applied thereon) may remain unchanged. Hence, computing the full graph for cycle **160** may be performed without investigating entanglement properties of all pairs of qubits.

[0096] In some exemplary embodiments, a quantum program may have a reversibility property, enabling to compute the initial state based on the last state. In some cases, the reversibility property may be utilized to compute information regarding a preceding cycle using information of a succeeding cycle. For example, given the information the distribution of qubit Q in cycle **160**, and assuming the function between cycle **150** and **160** that affects qubit Q is a simple function without quantum operations (f ( ), the distribution of qubit Q in cycle **150** may be computed based on the distribution of qubit Q in cycle **160** and using the function f-**1** ( ) As another example, entanglement graph of cycle **160** may be utilized to statically compute the entanglement graph of cycle **150**.

[0097] Referring now to FIG. **2** showing a block diagram of an apparatus, in accordance with some exemplary embodiments of the disclosed subject matter. The apparatus of FIG. **2** may be configured to perform the steps of the methods of FIGS. **1**A-**1**F.

[0098] In some exemplary embodiments, Apparatus **200** may comprise one or more Processor(s) **202**. Processor **202** may be a Central Processing Unit (CPU), a microprocessor, an electronic circuit, an Integrated Circuit (IC) or the like. Processor **202** may be utilized to perform computations required by Apparatus **200** or any of it subcomponents. It is noted that Processor **202** may be a traditional processor, and not necessarily, a quantum processor.

[0099] In some exemplary embodiments of the disclosed subject matter, Apparatus **200** may comprise an Input/Output (I/O) module **205**. I/O Module **205** may be utilized to provide an output to and receive input from a user, such as, for example to obtain debug commands and instructions from a user, retrieve debug instructions via network, receive instructions to start a new debugging sessions or continue an existing debugging session, provide output to the user, indicating execution

progress, values of variables of the quantum program, or the like.
[0100] In some exemplary embodiments, Apparatus **200** may comprise Memory **207**. Memory **207** may be a hard disk drive, a Flash disk, a Random Access Memory (RAM), a memory chip, or the like. In some exemplary embodiments, Memory **207** may retain program code operative to cause Processor **202** to perform acts associated with any of the subcomponents of Apparatus **200**. Memory **207** may comprise one or more components as detailed below, implemented as executables, libraries, static libraries, functions, or any other executable components.
[0101] In some exemplary embodiments, Quantum Debugger **210** may be configured to enable a user to perform debugging of a quantum program (not shown), that is executed on a quantum computer, such as Quantum Execution Platform **290**. The user may initiate debugging sessions in which, the user may set conditional and unconditional breakpoints in specific cycles, set conditional breakpoints regardless of a specific cycle (e.g., in all cycles), perform step forward and backwards operation, perform run operations, request to view a value of a variable, such as a qubit or set of qubits, request to view quantum properties, such as entanglement, density matrix, or the like, in a current cycle, or the like. Quantum Debugger **210** may be configured to provide the user with a user experience (UX) similar to that of a traditional debugger, in which the current state of the program is being tracked and commands affect the current state, such as by causing execution of additional instructions of the quantum program (e.g., step forward or run instructions) or by updating the memory by updating values of variables. Such user experience may be provided regardless of the implementation details of Quantum Debugger **210** in which the quantum program may not be paused in the middle of the execution and may not be resumed, in opposed to traditional debuggers.
[0102] In some exemplary embodiments, Execution Module **260** may be configured to execute the quantum program on the target quantum computer, Quantum Execution Platform **290**. In some exemplary embodiments, Execution Module **260** may be configured to execute the program once, and obtain the execution result, such as the output state provided by Quantum Execution Platform **290**. Additionally or alternatively, Execution Module **260** may be configured to execute the program a multiple times and obtain multiple output results. The results may then be aggregated together, such as by creating an averaged output state, calculating a distribution function, or the like, which may be considered to be the output of the compiled quantum circuit. In some exemplary embodiments, Execution Module **260** may be configured to execute the quantum program until reaching an intermediate state which is not the last, final state of the quantum program.
[0103] In some exemplary embodiments, Qubit Value Determinator **220** may be configured to determine a value of a qubit (or set of qubits) in a specific cycle. In some exemplary embodiments, Qubit Value Determinator **220** may be configured to utilize Execution Module **260** to execute the quantum program on Quantum Execution Platform **290** until reaching the specific cycle. In some exemplary embodiments, a plurality of executions may be performed, the number of which may be determined based on desired quality, available and allocated resources, user preferences, or the like. In some exemplary embodiments, the specific cycle may be the cycle that is displayed to the user as the current cycle of the quantum program in the debugging session. In some exemplary embodiments, Quantum Value Determinator **220** may be configured to compute a value of the qubit using quantum tomography. Additionally or alternatively, Value Determinator **220** may be configured to evaluate a property of the qubit (or set of qubits) such as entanglement therebetween, distribution of values, standard deviation of the value, or the like.
[0104] In some exemplary embodiments, Qubit Value Determinator **220** may be configured to determine the value of the qubit using a previously determined value of the qubit or other qubits in another cycle, which may be a preceding cycle or a succeeding cycle to the specific cycle. In some exemplary embodiments, it may be determined whether to attempt a static analysis to determine the value of the qubit or to perform dynamic analysis, in which the Execution Module **260** is invoked. In some exemplary embodiments, static analysis may be performed using Quantum Program

Analysis Module **240**. Quantum Program Analysis Module **240** may be configured to analyze the quantum program and determine a value or property of a qubit based on operations performed thereon between the cycle of interest and another cycle, for which information was previously computed.

[0105] In some exemplary embodiments, Qubit Value Propagator **230** may be configured to propagate, in a controlled manner, a desired target value into a qubit in a specific cycle, such as the cycle of the quantum program that is considered by Quantum Debugger **210** as the "current" cycle. In some exemplary embodiments, Qubit Value Propagator **230** may synthesize a transformative quantum program based on the value of the target qubit and based on the target value. The quantum program may be updated to cause the transformative quantum program to be executed after the target cycle. In some exemplary embodiments, a modified quantum program may be generated and reused for the remaining of the debugging session of Quantum Debugger **210**, so as to provide the illusion of the value of the qubit being modified in memory mid-execution.

[0106] In some exemplary embodiments, in case of backward execution traversal (e.g., step backward) by Quantum Debugger **210**, the transformative quantum program may or may not be removed. As an example, if the backward traversal does not backtrack before the cycle in which the new value is propagated, the same modified quantum program may be utilized. If the backward traversal goes to a cycle before the cycle in which the transformative quantum program is implemented, the original quantum program may be utilized. In some cases, a new quantum program may be created, in which the same controlled propagation operation is performed earlier. In some exemplary embodiments, a different transformative quantum program may be synthesized and utilized. Additionally or alternatively, the same transformative quantum program may be utilized. In some exemplary embodiments, a new program can be created in which some operations are reversed. The new quantum program may be composed of several code snippets that are concatenated one after the other: P.sub.1.i.Math.TQP.sub.i.Math.P.sub.i..j.sup.−1), where P.sub.1..i is the segment of the quantum program from its beginning and until cycle i, in which the propagation of the value was performed. TQP.sub.i is the transformative quantum program that was applied in cycle i. P.sub.i..j.sup.−1 is a quantum program that is an inverse of the quantum program from cycle i to cycle j (j<i), where j is the current cycle in the debugging session. In such a manner, Quantum Debugger **210** may be able to utilize the reversibility property of quantum programs so as to propagate the value in the correct location, and undo all operations other than the propagation operation. Additionally or alternatively, the quantum program that is executed may be P.sub.1..j.Math.TQP.sub.i, where P.sub.1..j is the segment of the quantum program from its beginning and until cycle j, the current cycle of the debugging session, and TQP.sub.i is the transformative quantum program that was applied in cycle i, and which is now applied after cycle j (j<i).

[0107] In some exemplary embodiments, Display Enhancer Module **250** may be configured to provide display to the user of Quantum Debugger **210**. In some exemplary embodiments, Display Enhancer Module **250** may dilute excessive and non-important information, such as negligible values in a distribution. In some exemplary embodiments, Display Enhancer Module **250** may be configured to display a graphical representation of an entanglement graph to the user. In some exemplary embodiments, the display may aggregate several qubits together, such as qubits that are used in tandem to represent a single variable (e.g., q.sub.1, q.sub.2, . . . , q.sub.n may be utilized together to represent a variable using n-qubits).

[0108] In some exemplary embodiments, Apparatus **200** may be utilized with respect to non-deterministic classic software, instead of with respect to quantum program. In some exemplary embodiments, instead of utilizing Quantum Execution Platform **290**, Execution Module **260** may utilize a classic execution platform (not shown). In some exemplary embodiments, Execution Module **260** may execute the program using Processor **202** and not using a separate execution platform. Additionally, or alternatively, Qubit Value Propagator **230** may be replaced by a module

configured to set a distribution value to a variable, Qubit Value Determinator **220** may be replaced by a module configured to determine a distribution value of a variable, Quantum Program Analysis Module **240** may be replaced be a classic program analysis module, or the like. In some exemplary embodiments, such embodiment may include a classic debugger instead of Quantum Debugger **210**, enabling the user to view distribution of variable values of the non-deterministic classic software, to modify the distribution of variable values, or the like.

[0109] Referring now to FIG. **3**A showing a flowchart diagram of a method, in accordance with some exemplary embodiments of the disclosed subject matter. It is noted that FIG. **3**A is applicable to both quantum programs and classic software. However, for the clarity of disclosure, the method is exemplified using a quantum program.

[0110] On Step **300**, Variables of Interest (VOI) in the program may be determined. The VOI may be determined based on user instruction. As an example, the user may provide a command requesting to view a value of a variable. Additionally, or alternatively, the user may utilize a debugger and issue a "print" command with respect to a variable, indicating that such variable is a VOI. In some exemplary embodiments, the determined VOI may include one variable (e.g., "print v"), several variables (e.g., "print v1+v2"), or the like.

[0111] On Step **310**, a Location of Interest (LOI) in the program may be determined. The LOI may be a specific location in the program, such as defined by a specific line of code, a specific cycle, a semantic location where a conditional breakpoint is active, or the like. In some exemplary embodiments, the LOI may be at the end of the program. Additionally, or alternatively, the LOI may be at any cycle that is not a terminating cycle. The LOI may be determined based on user instruction, such as user instruction to a debugger, based on the command issued by the user, or the like.

[0112] For the purpose of illustrating the disclosed subject matter, the operation of the method of FIG. **3**A is exemplified using a Quantum Program (QP) **400***a* that is illustrated in FIG. **4**A. As can be appreciated, Quantum Program **400***a* utilizes six qubits (q.sub.0 . . . q.sub.5). In Steps **300** and **310**, it may be determined that qubits q.sub.0 and q.sub.1 (**410**, **412**) are of interest in cycle 14 (e.g., values al **420**, **422**). As an example, the determination may be based on a user instruction to print the values of q.sub.0 and q.sub.1 in cycle 14. Additionally, or alternatively, the user instruction may be to print the values of q.sub.0 and q.sub.1 when a conditional breakpoint is active, and it may be determined that the breakpoint is active at cycle 14.

[0113] On Step **320**, a last location affecting the VOI before the LOI may be determined. In some exemplary embodiments, a last location before the LOI in which the value of any of the VOI is potentially set may be identified.

[0114] Referring to FIG. **4**A, as can be appreciated the values of any of the VOIs (**410**, **412**) is potentially affected on cycle 10 in view of Gate **430**. Gate **430** may be a logical quantum gate operating on both q.sub.1 (**412**) and q.sub.5. Hence, Gate **430** may potentially affect and modify the value of q.sub.1 (**412**). Accordingly, the last location affecting the VOI before the LOI is determined to be cycle 10. In some exemplary embodiments, all computations performed after cycle 10 (e.g., computations of cycles **11**-**14**) may be avoided, as such computations will not affect the investigated result.

[0115] On Step **330**, impact analysis may be performed on the VOI from the last location. In some exemplary embodiments, code that can potentially impact the VOI value at the last location is identified. In some exemplary embodiments, one impact analysis technique may be based on the idea of light cone. The impact analysis may be based on backward light cone. Gate A may be considered as outside the backward light-code of gate B, of the forward light-cone from B does not include gate A. A forward light-cone L may be defined as all gates with at least one input qubit causally connected to the output qubits of W.

[0116] In some exemplary embodiments, the impact analysis may be static impact analysis, dynamic impact analysis, a combination thereof, or the like. In some exemplary embodiments, the

impact analysis may unroll loops and other iterative code to determine impact of the code in different iterations. As an example, assuming that the software is a simulation that continues day by day. The value of variable x that is computed in each day, is computed based on the value of y at the preceding day. If, in this example, the LOI is the 10th day and VOI is x (i.e., we are interested in the value of x on day 10, it may be determined that y is of interest and is part of the VOI. However, on day 10, the value of y may be considered as not part of the VOI. By unrolling the iterative code, it may be determined that y is VOI while y.sub.10 is not, where y.sub.i is the value of y in the i-th day.

[0117] On Step **340**, code that does not impact the investigated result may be removed. The code may be identified based on the analysis of Step **330**. Referring now to FIG. **4**B, showing a modified Quantum Program **400***b* that is generated based on Quantum Program **400***a*. In view of the determination of Step **320**, Quantum Program **400***b* does not include any calculation performed after cycle 10 of Quantum Program **400***a*. Moreover, Gate **440** that connects q.sub.2 and q.sub.3 at cycle 9 of Quantum Program **400***a* is not in the light-cone of the VOI. Accordingly, such gate is removed from Quantum Program **400***b* (see **440***b*). As can be appreciated, in view of the modification, not only is a gate spared, a cycle may be spared as well, as no other computation is required to be performed in cycle 9. Accordingly, Quantum Program **400***b* may include only 9 cycles, instead of the original 15 cycles of Quantum Program **400***a*, 14 of which were supposed to be implemented in order to determine the investigated result of q.sub.0 and q.sub.1 at cycle 14. So, in this example, a reduction in more than 35% of the cycles is achieved.

[0118] On Step **350**, the updated program may be compiled. In some exemplary embodiments, Quantum Program **400***b* may be compiled by a compiler so as to provide a quantum program that can be executed on a specific type of target execution platform. In some exemplary embodiments, the compiler may predict small-scale functionality and replace it with a better implementation, as part of local optimizations. It is noted that the same quantum circuit may be compiled to a different implementation for different execution platforms, such as in view of the gates available in the execution platform which are used to implement the functionality of the gates defined in the quantum circuit. Hence, the compiler may create a different quantum circuit having an equivalent functionality to that of the quantum program generated on Step **340**, potentially varying in performance.

[0119] On Step **360**, the compiled updated program may be executed to receive results. The compiled updated program may be executed a plurality of times (e.g., N times). In some exemplary embodiments, the number of times to be executed may be determined a-priori, posteriori, or the like. In some exemplary embodiments, the user may manually define a number of times to execute the program. Additionally or alternatively, a quantitative measurement may be provided to determine, dynamically, if sufficient number of executions were performed. For example, if additional executions do not add significant amount of data, change the already gathered information, or the like, it may be determined that no additional executions are required. In some exemplary embodiments, the executions may be performed until reaching a desired quality level (e.g., confidence level in measurement, estimated statistical deviation, or the like). In some exemplary embodiments, the number of executions to be performed may be based on the resources available for such task.

[0120] In some exemplary embodiments, during the execution, the execution cost may be measured and compared with the predicted cost estimation. In some cases, if the optimization performed using the impact analysis (e.g., on Steps **330**-**340**) performs differently than expected (e.g., sub-performs with respect to the expectations), a different optimization may be performed instead of or in addition to the optimization that was already performed. The determination whether to perform such different optimization may be based on the estimated overhead costs and execution costs of additional optimizations. In some exemplary embodiments, the measurement of the actual overhead costs and execution costs of the modified program may be utilized for future predictions, thereby

potentially improving the quality of future predictions.
[0121] On Step **370**, the results from the multiple executions of the updated program (Step **360**) are aggregated and outputted. The results may be provisioned to the user, such as in reply to the user inquiry (e.g., user command "print q.sub.0, q.sub.1 @14" requesting to print the values of specific qubits (q.sub.0, q.sub.1) at a specific cycle (**14**)). The aggregated result may be the investigated result relating to the VOI at the LOI.
[0122] In some exemplary embodiments, the investigation may require the density matrix, or other measurements derivable therefrom, of the VOI at the LOI of the program. As can be appreciated, instead of utilizing the program itself, the disclosed subject matter may create an alternative program ("updated program") that has the same density matrix of the VOI at the LOI as of the original program. It is noted that the alternative program is constructed based on the original program and may, in some cases, be executed faster, may require a reduced amount of resources and qubits, or the like.
[0123] Referring now to FIG. **3**B, showing a flowchart diagram of a method, in accordance with some exemplary embodiments of the disclosed subject matter.
[0124] On Step **330**, the impact analysis is performed using a light-cone analysis. The light-cone analysis identifies all gates in the quantum program that are connected to the inputs of the VOIs at the LOI. In some exemplary embodiments, the light-cone analysis may depend on the type of gate that is connected to the VOI. In some exemplary embodiments, an iterative process of performing the light-cone analysis may comprise a procedure in which the gates that are directly affecting the VOI at the LOI are identified. Each such gate may be considered to be of interest and value of qubits at the relevant cycle may be added to the VOI for further analysis, until reaching the initial cycle of the quantum program. In some exemplary embodiments, each gate may be analyzed. If the gate is partial impacting gate, only potentially relevant qubits are added to the VOI (**338**). If the gate is a full impacting gate, all qubits that are connected to the full impacting fate may be added to the VOI (**336**). It is noted that the gate being partial impacting gate or full impacting gate may depend on the specific implementation of the quantum computer that would execute the quantum program.
[0125] As an example, consider Gate **430** of the quantum program of FIG. **4**A. Such a gate may be considered a full impacting gate. Hence, the light-cone analysis may deduce that the value of q.sub.1 in cycle 10 may be affected by the values of q.sub.1 and q.sub.5 in cycle 9.
[0126] As another example, Gate **450** may be a SWAP gate. Such a gate swaps the values of two qubits (in the illustration q.sub.1 and q.sub.4), and may be a partial impacting gate. As such, the value of q.sub.1 in cycle 9 may be affected by the value of q.sub.4 in cycle 8, but is not affected by the value of q.sub.1 in cycle 8. Accordingly, FIG. **4**C showing a modified Quantum Program **400***c* that is generated based on Quantum Program **400***a*. Quantum Program **400***c* further excludes Gate **470** (see **470***c*) and Gate **460** (see **460***c*). Accordingly, the number of cycles can also be reduced from 9 cycles in Quantum Program **400***b* to 8 cycles in Quantum Program **400***c*. As can be appreciated, cycle number 6 can be removed due to the removal of Gate **460**, while cycle number 3, from which Gate **470** was removed, can not be reduced, due to other gates operating in the same cycle.
[0127] Referring now to FIG. **5** showing a flowchart diagram of a method, in accordance with some exemplary embodiments of the disclosed subject matter.
[0128] On Step **500**, a classic program is obtained. The classic program may be provided in any programming language, such as C++ ™, Java™, assembly, or the like. In some exemplary embodiments, the classic program may not utilize qubits. The classic program may be configured to be executed by a classic, non-quantum, execution platform, such as utilizing a digital processor. In some exemplary embodiments, the classic program may be a non-deterministic software, such as a simulation of a complex environment that includes random or non-deterministic information, a software that is based on a stochastic model, or the like.

[0129] On Step **510**, an instruction to obtain a distribution of values of a variable at a location may be obtained. For example, a programmer may utilize a programming instruction to request that at a specific location, the program would return the distribution of values or otherwise gather such information to determine a desired output. In some exemplary embodiments, the information may be gathered and retained for logging purposes, for analytic purposes, for purposes of validating an underlying model of the classic program, or the like. As another example, the programmer may utilize a debugger to debug the program and may instruct the debugger to provide such information. In some exemplary embodiments, the user may set a breakpoint or otherwise traverse to a specific program location and request such information in the desired location. As another example, the location may be defined using a conditional breakpoint, such that the location is not defined exclusively using a program instruction value, but also combines a semantic condition. In some cases, only a semantic condition may be utilized without relating to a specific location (e.g., whenever the variable “var” is assigned the value “0”).

[0130] The classic program may be executed a plurality of times (**520**-**530**), and the results relevant to the query defined in Step **510** may be aggregated and displayed to the user (Step **540**). In some exemplary embodiments, the program may be executed as is. Additionally or alternatively, optimization may be performed, such as to cut-out any code that is irrelevant to the query to speed-up execution. Additionally or alternatively, the number of times to execute the program may be a-priori defined, posteriori determined, or the like. In some exemplary embodiments, the executions may be performed until the aggregated results are within a desired quality level.

[0131] Referring now to FIG. **6** showing a flowchart diagram of a method, in accordance with some exemplary embodiments of the disclosed subject matter. FIG. **6** exemplifies the usage of the controlled propagation operation in accordance with the disclosed subject matter. A program may be obtained (**600**) and a controlled propagation instruction may be provided (610). Accordingly, the program may be executed a plurality of times (**620**), the number of which may be determined a-priori, posteriori, or the like. During each execution, the value of the variable is a concrete value. The concrete value may be non-deterministically selected at the location in which the controlled propagation instruction is implemented, causing the propagation of the distribution value. Additionally or alternatively, the value may be propagated at the location in which the value is examined or utilized. The value may be propagated directly based on the distribution, such as by sampling a value from the distribution. In such an embodiment, during each execution, a potentially different specific value is given, while adhering to the distribution.

[0132] One technical problem dealt with by the disclosed subject matter is enhancing the processing capabilities of data obtained from quantum sensors.

[0133] In some exemplary embodiments, quantum sensors may be configured to sense, measure, or the like, quantum properties of a sensed physical phenomenon, physical system, physical object, or the like. For example, a quantum sensor may be configured to measure a quantum state of a physical phenomenon. In some exemplary embodiments, quantum sensors may be sensitive to quantum properties such as a magnetic field of atom, a location of an atom, a speed of an atom, a magnetic field of a molecule, a location of a molecule, a speed of a molecule, a quantum state of a physical situation, quantum information processing, quantum information measuring, or the like.

[0134] In some exemplary embodiments, quantum sensors may demonstrate superiority over classical sensors, e.g., as disclosed in Hsin-Yuan Huang et al. Quantum advantage in learning from experiments. arXiv: 2112.00778 (2022). arxiv.org/abs/2112.00778, which is hereby incorporated by reference in its entirety for all purposes without giving rise to disavowment. For example, the accuracy level of quantum sensor measurements may be higher than the accuracy level of classical sensor measurements.

[0135] In some exemplary embodiments, a quantum sensor may store a measurement of one or more quantum properties as a quantum state. In some exemplary embodiments, the information stored in quantum sensors, e.g., the quantum state, may be loadable on a quantum computer, e.g., as

disclosed in Vorobyov, V., Zaiser, S., Abt, N. et al. Quantum Fourier transform for nanoscale quantum sensing. npj Quantum Inf 7, 124 (2021). doi.org/10.1038/s41534-021-00463-6, which is hereby incorporated by reference in its entirety for all purposes without giving rise to disavowment.
[0136] In some exemplary embodiments, it may be challenging to transfer the quantum state that was measured by the quantum sensor over classical channels, such as in order to evaluate, recover, or restore the quantum state at a receiving entity (e.g., a classical or quantum computer). In some exemplary embodiments, the quantum state that was measured by the quantum sensor may not be directly measurable. In some exemplary embodiments, in order to evaluate, in classical terms, quantum data that is measured by a quantum sensor, the quantum data must be measured a large number of times (e.g., more than a threshold, such as thousands or millions of times) by the quantum sensor. For example, each measurement of the quantum sensor may or may not require to reset the measured physical system, object, phenomenon, or the like, to their initial state.
[0137] In some exemplary embodiments, in order to process quantum data that is measured by a quantum sensor, measurements of the quantum data may be performed and records of measurement results may be communicated to a classical or quantum computer. In some cases, the results of each measurement may be sampled, such as using tomography measurements, in order to assess the measured quantum data. In some exemplary embodiments, sampling of each execution may comprise performing a large number of measurements (e.g., more than a threshold), in order to obtain a statistically significant result, to account for statistical fluctuations and errors, or the like. In some cases, the results of each measurement, samples thereof, or the like, may be stored and communicated to a classical or quantum computer.
[0138] In some exemplary embodiments, this process of evaluating and/or communicating the quantum state using many executions and measurements may be highly consuming of resources, e.g., time resources, computational power, storage resources, resources for resetting the state of the measured physical system, or the like. It may be desired to overcome such drawbacks, and reduce the number of resources that are required for the evaluation process. For example, it may be desired to reduce a number of measurements of sensed quantum data that is required for estimating the quantum state.
[0139] Another technical problem dealt with by the disclosed subject matter is enhancing an effectiveness of storing data that is obtained from quantum sensors. In some exemplary embodiments, the amount of classical storage that is needed for capturing an arbitrary quantum state may be exponential in the number of quantum bits (qubits) of the arbitrary quantum state. It may be desired to overcome such drawbacks, and reduce the amount of classical memory storage that is needed for representing a sensed quantum state.
[0140] Yet another technical problem dealt with by the disclosed subject matter is enhancing communication capabilities of data obtained from quantum sensors. For example, it may be desired to communicate a sensed quantum state that is sensed by a quantum sensor, via a classical communication channel, to one or more classical or quantum computers. In some exemplary embodiments, since the amount of classical storage that is needed for representing a quantum state may be exponential in the number of qubits of the quantum state, the conveying of information from quantum sensors to classical computers may be inherently challenging. In some exemplary embodiments, it may be desired to communicate a sensed quantum state to classical computers, remote quantum computers that may not have quantum communication with the quantum sensor, or the like, in a manner that is not resource-consuming.
[0141] One technical solution provided by the disclosed subject matter is processing sensed quantum data, in a manner that allows for a reconstruction of an approximation of the sensed quantum data at later times, at different computing platforms, or the like.
[0142] In some exemplary embodiments, a quantum sensor may measure a quantum state of an inspected phenomenon, system, object, or the like. In some exemplary embodiments, in order to process the sensed quantum state, the sensed quantum state that was measured by the quantum

sensor may be loaded and fed to a set of one or more qubits of a quantum computer. In some exemplary embodiments, the quantum computer may be connectable to the quantum sensor, which may enable the quantum computer to receive the quantum state from the quantum sensor. For example, the quantum computer may be physically wired or connected to the quantum sensor, or may connect to the quantum sensor in a wireless manner, such as by entanglement.

[0143] In some cases, the quantum computer may or may not comprise a relatively small computer, that corresponds to an output size of the quantum sensor. For example, the quantum computer may comprise a computer having a minimal number of qubits that is necessary for holding the sensed quantum state from the quantum sensor, having twice the quantity of the minimal number of qubits, having thrice the quantity of the minimal number of qubits, having any other number of times the quantity of the minimal number of qubits, having the quantity of the minimal number of qubits and an additional set of one or more auxiliary qubits, or the like.

[0144] In some exemplary embodiments, the set of one or more qubits that hold the sensed quantum state may belong, or be allocated, to a quantum circuit, e.g., a parametric quantum circuit. In some exemplary embodiments, the parametric quantum circuit may be designed for determining the sensed quantum state, evaluating the sensed quantum state, processing the sensed quantum state, or the like. In some exemplary embodiments, the parametric quantum circuit may be designed to comprise the set of qubits on which sensed data is loaded from the quantum sensor. For example, the parametric quantum circuit may be designed to manipulate a plurality of qubits, including at least the set of qubits holding the sensed state, over a plurality of cycles using a plurality of quantum gates.

[0145] In some exemplary embodiments, the parametric quantum circuit may comprise a quantum circuit in which at least some of the gates have parameters that are initially unspecified, constitute variables, or the like. In some exemplary embodiments, allowing certain gates to have tunable parameters may introduce flexibility to the parametric quantum circuit. In some exemplary embodiments, these parameters may be adjustable, enabling to optimize the circuit's performance for a specific quantum computation task. In some exemplary embodiments, the parameters may or may not be represented or included in the sensed quantum state from the quantum sensor.

[0146] In some exemplary embodiments, the parametric quantum circuit may comprise a set of quantum gates that evolve an initial quantum state at one or more initial cycles of the set of qubits, to a final state encoding an output of the circuit. For example, the final state may comprise a state of one or more output qubits, a state of the set of qubits at an end of the circuit's execution, or the like. In some exemplary embodiments, the initial quantum state of the set of qubits may be provided by the quantum sensor, and may correspond to the sensed quantum state measured by the quantum sensor. For example, the set of one or more qubits may be set, by the quantum sensor, to represent the quantum state at an initial cycle of the parametric quantum circuit, as the initial state of the set of one or more qubits.

[0147] In some exemplary embodiments, the parametric quantum circuit may be designed to manipulate the initial state with one or more sub-circuits, gates, quantum operations, or the like. In some exemplary embodiments, the parametric quantum circuit may be designed to comprise an inverse ansatz parametric circuit, a detection circuit, or the like, which may be configured to manipulate the initial state. For example, the inverse ansatz parametric circuit and the detection circuit may comprise subsequent sub-circuits of the parametric quantum circuit, respectively.

[0148] In some exemplary embodiments, the inverse ansatz parametric circuit may comprise an inverse of an ansatz parametric circuit. For example, applying both the inverse ansatz parametric circuit and the ansatz parametric circuit on a quantum state may be configured to result with the quantum state (e.g., when ignoring error rates), due to their inverse relationship. In some exemplary embodiments, an ansatz parametric circuit may represent, or implement, an ansatz (e.g., a hypothesis or educated guess) associated with the sensed quantum state. In some exemplary embodiments, the ansatz may be determined and utilized to capture or approximate the sensed

quantum state. For example, in case the quantum sensor measures energy of a molecule, the ansatz may represent a hypothesis that attempts to approximate the energy of the molecule. In some exemplary embodiments, the ansatz parametric circuit implementing the ansatz may comprise a circuit that, when executed, is configured to output an approximation of the sensed quantum state, in accordance with the ansatz. In some exemplary embodiments, an inverse ansatz parametric circuit implementing the inverse of the ansatz may comprise a circuit that, when executed, is configured to output an approximation of an inverse of the sensed quantum state. For example, in case the approximation of the sensed quantum state is a value of 4, the approximation of the inverse of the sensed quantum state may be a value of −4.

[0149] In some exemplary embodiments, the ansatz may belong to a parameterized family of quantum circuits that can be adjusted by tuning the set of parameters. In some exemplary embodiments, the ansatz may be associated to a set of parameters. In some exemplary embodiments, the set of parameters may comprise any property or parameter that can affect the circuit, such as parameters defining angles of rotation gates, which gate to include or exclude, or the like. In some cases, parameters of the ansatz may correspond to ones disclosed in Harper R. Grimsley et al. An adaptive variational algorithm for exact molecular simulations on a quantum computer. arXiv: 1812.11173. Nature Communications 10, 3007 (2019). arxiv.org/abs/1812.11173, which is hereby incorporated by reference in its entirety for all purposes without giving rise to disavowment.

[0150] For example, the sensed quantum state may correspond to a state or value of x, and an ansatz may provide a hypothesis that the sensed quantum state can be approximated by an integer. According to this example, the set of parameters may comprise parameters representing integer values, and the parameters may be tuned with valuations of different integer values (e.g., values of 0, 1, 2, 15, 200, or the like). In some cases, the ansatz may be represented by a circuit (e.g., the ansatz parametric circuit), by a set of parameters, or the like.

[0151] In some exemplary embodiments, one or more types of ansätze may be determined. In some exemplary embodiments, in case a physical system is measured by the quantum sensor, a physics-oriented ansatz may be formed from quantum gates that are associated to the physical system. For example, a physics-oriented ansatz may comprise encodings of elements of the physical system Hamiltonian. As another example, a physics-oriented ansatz may comprise encodings of a generic molecular or atomic orbitals. In some exemplary embodiments, a hardware-oriented ansatz may be formulated as a random and maximally condensed quantum circuit that can be efficiently executed on a quantum computer, e.g., in terms of cycle-wise depth, error rates, costs, gate count, or the like. In some exemplary embodiments, any other types of ansätze and associated parameters may be used, e.g., similar to the ansätze and parameters disclosed in M. Cerezo, et al. Variational Quantum Algorithms. arXiv: 2012.09265. Nature Reviews Physics 3, 625-644 (2021), which is hereby incorporated by reference in its entirety for all purposes without giving rise to disavowment (hereinafter referred to as ‘M. Cerezo’).

[0152] In some exemplary embodiments, the ansatz may be predicted or estimated to have at least one set of valuations to the set of parameters, that, when applied to the parameters, provides an approximation of the sensor's data. In some exemplary embodiments, an approximation of the sensor's data may be considered acceptable when it closely aligns with the actual sensed data, adhering to a predefined threshold. For example, the predefined threshold may represent the maximum allowable deviation or difference between the approximated states and the true sensor measurements. In some exemplary embodiments, the approximation may be required to fall within a defined acceptable range, ensuring a level of accuracy deemed satisfactory.

[0153] In some exemplary embodiments, the set of parameters of the ansatz may or may not correspond to properties of the measured phenomenon that were measured by the quantum sensor. For example, the quantum sensor may measure a partial state of a molecule, and the set of parameters may define a full state of the molecule. As another example, the quantum sensor may

measure a first set of parameters representing a state of a molecule, while the set of parameters of the ansatz may measure a second disjoint set of parameters representing the same state of the molecule.

[0154] In some exemplary embodiments, in order to measure whether or not a set of one or more valuations provides an acceptable approximation of the sensed quantum state, an inverse of the ansatz may be applied as the inverse ansatz parametric circuit on the initial state of the set of qubits. In some exemplary embodiments, in case the inverse ansatz parametric circuit corresponds to an inverse of the sensed quantum state, the processed state may correspond to the ground state (the |0) state), the zero state, a near-zero state (the probability of measuring zero from the output being above a certain probability), or the like. In case the resulting state is zero or nearly zero, the ansatz may be determined to correspond to the sensed quantum state, and thus the ansatz may be verified and determined to be correct. In such cases, the valuation of the ansatz that resulted with the zero or near zero measurement, may be stored and utilized to determine the sensed quantum state.

[0155] In some exemplary embodiments, the inverse ansatz parametric circuit may be configured to obtain the set of one or more qubits holding the sensed quantum state as input qubits, and output, based thereon, a processed state that is obtained by manipulating the quantum state with one or more gates. In some exemplary embodiments, the inverse ansatz parametric circuit may or may not obtain one or more additional qubits such as auxiliary qubits that are not included in the set of qubits. For example, in case the set of valuations provides an approximation of the sensed quantum state, applying the inverse ansatz parametric circuit on the set of qubits will result with at least a subset of the set of qubits holding a state of zero, near zero, or the like.

[0156] It is noted that when relating to the set of qubits having a zero or near zero state, the disclosed subject matter is not limited to all qubits of the set of qubits having a zero or ground state (the |0) state). For example, a subset of measurements of the set of qubits may result with a zero state, with a state that is nearly zero, or the like. In some exemplary embodiments, a state may be considered nearly zero in case the state adheres to a maximum allowable deviation from the zero state (e.g., according to a threshold). In some exemplary embodiments, states of the set of qubits may be sampled over a plurality of circuit executions, measurements, or the like, and their states may be considered to be zero or near zero states based on an average of different measurements of each qubit, a majority vote, or the like.

[0157] As an example, a sensed quantum state may correspond to a state of x, and may be loaded to the set of qubits. A user may not have access to the loaded state, and thus the sensed state may be measured indirectly using an ansatz. According to this example, an ansatz may be determined, and may predict that the sensed quantum state can be approximated by an integer value. The ansatz may be utilized for designing, or constructing, an inverse ansatz parametric circuit associated with a parameter p being an integer. For example, the inverse ansatz parametric circuit may subtract the value of p from one or more input qubit states. The parameter p may be instantiated with various integer values. Upon instantiating p with the value of 3, which is a close approximation of the state, the inverse ansatz parametric circuit may output the set of qubits with a processed state of zero or near zero. For example, instantiating p with the value of 3 may result with a greatest number of zero states of the set of qubits compared to any other valuation of the parameter p. In other cases, the sensed quantum state may correspond to any other quantum state, such as a highly entangled state, a pure state, or the like.

[0158] In some exemplary embodiments, the parametric quantum circuit may or may not be designed to comprise a detection sub-circuit, which may be configured to indicate a distance measure between the processed state outputted from the inverse ansatz parametric circuit, and between a predetermined state (e.g., the zero state). For example, in case the predetermined state is zero, the detection sub-circuit may comprise a zero detection sub-circuit, configured to determine whether or not applying the inverse ansatz parametric circuit on the initial state results with a zero

or near zero state. In some exemplary embodiments, setting the predetermined state to be zero, may be advantageous, at least since it may increase an efficiency of executing the parametric quantum circuit compared to setting the predetermined state to any other state, value, number, or the like. In some cases, the predetermined state may be set to any other number, value, quantum state, or the like.

[0159] In some exemplary embodiments, the detection sub-circuit may be configured to obtain the processed state from the set of one or more qubits, and output one or more output states indicating whether or not the processed state is zero, near zero, or the like. In some cases, the output states may indicate a distance between the processed state and a predetermined state such as zero. In some exemplary embodiments, the detection sub-circuit may output a single output state via a single qubit, may output one or more output states via two or more qubits, or the like. For example, the detection sub-circuit may be configured to output a single zero state in case that the processed states of the set of qubits are determined to be zero or near zero, and to otherwise output a different state such as a state of one. As another example, the detection sub-circuit may be configured to output a state of one in case that the processed states of the set of qubits are determined to be zero or near zero, and to otherwise output a state of zero. In other cases, the detection sub-circuit may be configured to indicate the distance between the processed state and the predetermined state in any other way.

[0160] In some cases, the detection sub-circuit may be configured to measure the distance between the processed state and the predetermined state based on an average of different measurements of each qubit, a majority vote, or the like. In some exemplary embodiments, the processed state provided from the inverse ansatz parametric circuit may be determined by the detection sub-circuit to be near zero, such as in case that a certain percentage of samples of measurements of the set of qubits results with zero, near zero, or the like. In some exemplary embodiments, the assessment of the nullness of the processed states may be achieved by any means of verification. For example, the assessment of the nullness of the processed states may correspond to one or more methods disclosed in US Patent Publication Number 20230409953, entitled “Auxiliary Qubit Verification in Quantum Circuits”, filed on May 24, 2022, which is hereby incorporated by reference in its entirety for all purposes without giving rise to disavowment.

[0161] In some cases, the detection sub-circuit may be omitted from the parametric quantum circuit, and the distance measure between the processed state and between the predetermined state may be determined directly from the output of the inverse ansatz parametric circuit, e.g., using tomography measurements. For example, all states of output qubits from the inverse ansatz parametric circuit may be measured to determine the probability that they indicate a state of zero, and obtaining a statistically significant result may require to execute the parametric quantum circuit a large number of times, e.g., thousands or millions of times. In some cases, in case the processed state is a computational basis (e.g., composed purely of zeros and/or ones), the detection sub-circuit may be omitted, and vice versa. It is noted that the sensed quantum state may not constitute a computational basis. In some cases, using a detection sub-circuit may enable to reduce resources used for measuring the processed state, enhance an accuracy of the measurements, or the like.

[0162] In some exemplary embodiments, the valuations of the set of parameters associated with the inverse ansatz parametric circuit may be adjusted iteratively as part of a Variational Quantum Algorithm (VQA), based on an output from the detection sub-circuit. In some exemplary embodiments, a VQA may constitute a quantum algorithm class that may be used for solving optimization problems. In some exemplary embodiments, VQAs may involve an iterative process where a quantum circuit is constructed, executed, and measured; the measurement results may determine the construction of the next iterated circuit. In some exemplary embodiments, the iterations may be executed for different logical variations of logical parameters associated with the parametric quantum circuit. For example, VQA algorithms are disclosed in M. Cerezo. In some cases, VQA schemes may be designed to leverage quantum computers to address problems that

classical computers find challenging, such as problems related to optimization and machine learning. In some exemplary embodiments, VQA schemes may enable to exploit quantum parallelism during the evolution of the quantum state.

[0163] In some exemplary embodiments, a loop of VQA may be implemented to iteratively adjust valuations for the set of parameters associated with the inverse ansatz parametric circuit. In some exemplary embodiments, each iteration, the output state from the detection sub-circuit may be determined, estimated, measured, or the like, and these measurements may be used, by a classical processor, to update the parameters of the inverse ansatz parametric circuit iteratively, until the algorithm converges to a solution that minimizes an objective function associated with the optimization problem (e.g., resulting with a zero or near zero state).

[0164] In some exemplary embodiments, each VQA iteration, the valuations for the set of parameters may be selected or determined to include values in a defined order, random values, values selected by a greedy search algorithm, values selected by a non-greedy search algorithm, or the like. For example, at each iteration, a classical optimization may be carried out with respect to a value function that aims to nullify the final result of the quantum computation. The optimization may be designed to make the resulting quantum state as close as possible to the zero state.

[0165] In some exemplary embodiments, for each valuation of the set of parameters, the quantum computer may be configured to execute the parametric quantum circuit using the selected valuation. In some exemplary embodiments, the quantum computer may be configured to execute the parametric quantum circuit a plurality of times, such as in order to accurately measure the output quantum state.

[0166] In some exemplary embodiments, a quantum measurement by the quantum sensor may be at least weakly destructive, causing a destruction of at least part of the measured quantum state. In some exemplary embodiments, since VQA is an iterative process, and any quantum measurement is at least weakly destructive, the possibility of changing the measured value at each iteration may be addressed. In some cases, a measurement of a phenomenon in the physical environment may require to reset the phenomenon or its state every measurement of the quantum sensor. For example, in case the measured property is highly quantum and the measurement is not sufficiently weak, the quantum system being measured may be required to be reset before each measurement, such as in order to reconstruct the state being measured. In other cases, a plurality of measurements of the phenomenon may be performed without resetting the phenomenon, such as in case that the destruction is not significant for the result. For example, in case the measured property is classical or semi-classical, or in case the measurement is weak enough, there may be little change of the measured quantity between each iteration, making resetting of the property redundant and not necessary.

[0167] In some exemplary embodiments, at the end of the iterative process of the VQA, output data may be generated. In some exemplary embodiments, the output data may comprise a representation or an indication of an approximation of the sensed quantum state, which may be obtained and stored compactly. In some exemplary embodiments, the representation of the quantum state may comprise a compressed or low-resource version of the quantum state approximation, that enables a reconstruction of the quantum state approximation at one or more computing devices.

[0168] In some exemplary embodiments, the output data may comprise a representation of the parametric quantum circuit, a representation of the inverse ansatz parametric circuit, a representation of the ansatz parametric circuit, a set of ansatz parameters associated parametric quantum circuit, a selected valuation of the ansatz parameters, an indication of the type or properties of the parametric quantum circuit, a value reconstruction circuit that is configured to reconstruct the parameter values or the quantum state approximation, a combination thereof, or the like. For example, the output data may comprise a compressed representation of a value reconstruction circuit that, when executed on a quantum computer, or simulated by a classical simulator of a quantum computer, will generate the approximation of the quantum state.

[0169] In some cases, the output data may not comprise a representation of the parametric quantum circuit itself, but rather a representation of the inverse ansatz parametric circuit that excludes the loaded quantum state. In some exemplary embodiments, instead of generating a representation of the entire parametric quantum circuit, a representation of the inverse ansatz may be generated. In some exemplary embodiments, the inverse ansatz parametric circuit may enable to reconstruct the quantum state approximation, such as along with the ansatz parameters, a corresponding predetermined state of the detection sub-circuit, or the like. For example, the inverse ansatz parametric circuit may be provided along with one or more predetermined states, corresponding valuations, or the like. In some cases, the output data may comprise a representation of the ansatz parametric circuit, instead of the inverse ansatz parametric circuit.
[0170] In some cases, the output data may not comprise a representation of the parametric quantum circuit itself, a sub-circuit thereof, or the like, and instead, the output data may comprise the set of ansatz parameters (with the respective valuations, predetermined states of the detection sub-circuit, or the like). In some exemplary embodiments, the parameters of the ansatz may be provided as a compressed version of the parametric quantum circuit or portions thereof, and may enable to reconstruct the approximation of the sensed quantum state. For example, in case the inverse ansatz parametric circuit removes a value of five from the loaded quantum state, the output data may be set to comprise the parameter value of five, the predetermined state of zero, or the like.
[0171] In some exemplary embodiments, the output data may be a compressed version, that uses less storage and communication capacity, compared to conducting tomography measurements of the sensed system directly. In some exemplary embodiments, the output data may be a compressed version, that uses less storage and communication capacity, compared to conducting and storing records of a plurality of measurements of the sensed quantum state. In some exemplary embodiments, the output data may be obtained using exponentially less measurements and samples compared to conducting tomography measurements of the sensed system directly.
[0172] In some exemplary embodiments, the output data may be communicated to a classical computer, a remote quantum computer that is not entangled with the quantum computer that executed the parametric quantum circuit, a classical computing cloud, a quantum computing cloud, a quantum execution platform, or the like. In some exemplary embodiments, the output data may be communicated via a classical communication medium such as long-distance communication, short distance communication, wireless communication, wired communication, or the like. For example, the output data may be communicated via WIFI™ communication. In some exemplary embodiments, an output module may be configured to provide the output data, an indication of the compressed data, or the like. For example, the output module may be comprised by the quantum computer over which the parametric quantum circuit was executed, and the quantum computer may provide the output data via one or more mediums to a separate and/or remote device.
[0173] For example, the output data may be communicated to a remote quantum computer. According to this example, the quantum state may be compactly represented by the ansatz parameters, enabling the remote quantum computer to reconstruct an approximation of the quantum state by executing a corresponding ansatz parametric circuit with the valuations of the ansatz parameters. According to this example, the remote quantum computer may generate the ansatz parametric circuit locally, obtain the ansatz parametric circuit from a third party, obtain the ansatz parametric circuit from the quantum computer that is attached to the quantum sensor, obtain the ansatz parametric circuit from a user such as a programmer, or the like.
[0174] As another example, the output data may be communicated to a classical computer. According to this example, since classical computers may not be able to execute quantum circuit, the data may be absent of any circuit representation, and may instead include solely ansatz parameters.
[0175] One technical effect obtained by the disclosed subject matter is enabling to process sensed quantum states, and communicate them to classical computers, remote quantum computers, or the

like.
[0176] Another technical effect obtained by the disclosed subject matter is enabling to communicate and process sensed quantum states while reducing computational resources, communication overhead, storage resources, or the like. In some exemplary embodiments, the reduction of resources may be exponential to the number of samples of the quantum state. In some exemplary embodiments, instead of storing a quantum state on classical parameters, which may be resource consuming, an approximation of the quantum state may be generated and represented by one or more classical parameters. For example, in case the sensed quantum state is loaded on 20 qubits, 2,097,150 (or 2{circumflex over ( )}21-2) classical parameters may be required to load the same state, while using the disclosed subject matter, one or more classical parameters may be used (e.g., the set of parameters of the inverse ansatz parametric circuit).
[0177] Yet another technical effect obtained by the disclosed subject matter is enabling to process sensed quantum states using a compact quantum computer. In some cases, a quantum computer that executes the parametric quantum circuit may or may not comprise a relatively small computer, that corresponds in size to a size (in terms of qubits) of an output of the quantum sensor. In some cases, in case the quantum sensor is not connected to a quantum computer, the sensed quantum state of the quantum sensor may be sampled directly from the quantum sensor, which may require a resource consuming tomography measuring process, and may thus be suboptimal.
[0178] The disclosed subject matter may provide for one or more technical improvements over any pre-existing technique and any technique that has previously become routine or conventional in the art. Additional technical problem, solution and effects may be apparent to a person of ordinary skill in the art in view of the present disclosure.
[0179] Referring now to FIG. **7**, showing a schematic block diagram, in accordance with some exemplary embodiments of the disclosed subject matter.
[0180] In some exemplary embodiments, Block Diagram **700** may depict blocks, each of which represents a component, stage, or subsystem of the disclosed subject matter, while interconnections between the blocks may illustrate how these components interact or are related in terms of functionality or information flow.
[0181] As depicted in FIG. **7**, a flow of Block Diagram **700** starts with Quantum Sensor **702** measuring a quantum state, and loading the quantum state to a set of qubits of Quantum Circuit **704**. For example, Loaded Sensor Data **741** may represent the set of qubits holding the quantum state. In some exemplary embodiments, Loaded Sensor Data **741** may comprise a set of qubits that are initialized with the quantum state at a start of Quantum Circuit **704**, which are subsequently manipulated by Inverse Ansatz **743** or State Detection **745**. In some exemplary embodiments, Loaded Sensor Data **741** may be absent of any logical gates.
[0182] In some exemplary embodiments, when executing Quantum Circuit **704**, Loaded Sensor Data **741** may be manipulated by Inverse Ansatz **743**. In some exemplary embodiments, Inverse Ansatz **743** may implement an inverse of an ansatz attempting to approximate the sensed quantum state that was measured by Quantum Sensor **702**. In some exemplary embodiments, Inverse Ansatz **743** may apply one or more quantum operations to Loaded Sensor Data **741**, and generate a processed version of the quantum state, which may be fed to State Detection **745**. In some exemplary embodiments, State Detection **745** may determine whether the processed data from Inverse Ansatz **743** is a predefined state such as zero, near zero, or the like. For example, State Detection **745** may output an indication of the nullness of the processed data.
[0183] In some exemplary embodiments, Loaded Sensor Data **741**, Inverse Ansatz **743**, and State Detection **745**, may or may not differ in the number of qubits utilized thereby. For example, while Loaded Sensor Data **741** and State Detection **745** may comprise a same number of qubits (although they may not necessarily be identical qubits), Inverse Ansatz **743** may in some cases manipulate a greater number of qubits than Loaded Sensor Data **741**, a same number of qubits as Loaded Sensor Data **741**, or the like. For example, Inverse Ansatz **743** may utilize additional auxiliary qubits that

are not used by Loaded Sensor Data **741** or State Detection **745**.
[0184] In some exemplary embodiments, Quantum Circuit **704** may be executed by Execute **706**, and its results may be measured by Measure **708**. For example, Execute **706** may execute Quantum Circuit **704** once, a plurality of times, or the like. In some cases, a phenomenon measured by Quantum Sensor **702** may or may not be reset every execution, every measurement of Quantum Sensor **702**, every defined number of measurements of Quantum Sensor **702**, every defined number of executions of Quantum Circuit **704**, or the like. In some cases, Measure **708** may measure an output state of the execution using tomography measurements, or using any other technique. For example, Measure **708** may measure the nullness indication of State Detection **745**.
[0185] In some exemplary embodiments, a VQA scheme may be implemented to iteratively adjust parameter values of Inverse Ansatz **743**, according to outputs from State Detection **745**. For example, the parameters of Inverse Ansatz **743** may be adjusted as part of the VQA scheme, thereby providing different valuations to the parameters every iteration. It is noted that a single iteration may comprise a single execution of Quantum Circuit **704**, a plurality of executions of Quantum Circuit **704**, or the like. In some exemplary embodiments, every iteration, the parameters of Inverse Ansatz **743** may be adjusted, such as according to the output from State Detection **745**. In some exemplary embodiments, the output from Inverse Ansatz **743** or State Detection **745** may be used, by a classical processor, to update the parameters of Inverse Ansatz **743** iteratively, until State Detection **745** detects a state of zero or near zero, until Inverse Ansatz **743** output a zero or near zero state, or the like.
[0186] In some exemplary embodiments, in case State Detection **745** indicates a result different from zero or near zero, the parameter values of Inverse Ansatz **743** may be adjusted. Otherwise, Output Data **710** may be generated. For example, Output Data **710** may be generated to comprise parameter values of Inverse Ansatz **743** that caused State Detection **745** to detect a state of zero.
[0187] Referring now to FIG. **8**, showing an exemplary flowchart diagram of a method, in accordance with some exemplary embodiments of the disclosed subject matter.
[0188] On Step **810**, a quantum state measured by a quantum sensor may be loaded on a set of one or more qubits of a quantum computer. In some exemplary embodiments, the quantum sensor may be configured to measure a quantum state of a physical phenomenon, object, or the like. For example, the quantum sensor may be configured to measure a location of an atom, e.g., spatial coordinates of the atom at the time of measurement.
[0189] In some exemplary embodiments, the quantum computer on which the quantum state is loaded, may be connectable to the quantum sensor via one or more mediums, e.g., physical connections, wireless connections, or the like, which may be quantum or classical. In some cases, the quantum computer may comprise a minimal-sized quantum computer, in terms of qubits, that is capable of implementing the method of Steps **810**-**850** for the quantum sensor. In other cases, the quantum computer may comprise any other sized computer, having any other number of qubits. In some exemplary embodiments, one or more mediums may enable the quantum computer to receive the quantum state from the quantum sensor.
[0190] In some exemplary embodiments, the quantum computer may receive the quantum state by loading the quantum state on a set of one or more qubits of the quantum computer. In some exemplary embodiments, the quantum computer may comprise at least the set of qubits over which the quantum state is loaded. For example, the quantum state may be loaded on a subset of the qubits of the quantum computer, on all of the qubits of the quantum computer, or the like.
[0191] In some exemplary embodiments, the set of one or more qubits that is loaded with the quantum state, may be part of a parametric quantum circuit. In some exemplary embodiments, the parametric quantum circuit may comprise one or more qubits that are set, by the quantum sensor, to represent the quantum state at an initial cycle of the parametric quantum circuit. In some exemplary embodiments, the quantum state may be loaded on a set of one or more qubits as an initial state of a quantum circuit.

[0192] In some cases, the quantum sensor and quantum computer may be housed in a single physical device, in separate physical devices, or the like. In case they are housed in a single physical device, the quantum computer may constitute an on-sensor embedded quantum computer. For example, an on-sensor embedded quantum computer may comprise a quantum computer that is embedded directly onto a quantum sensor. According to this example, the integration of the quantum computer with the quantum sensor may enhance a data acquisition process of the sensed quantum state. In other cases, the quantum sensor and quantum computer may be housed in separate devices, and connected via a medium such as a wire.

[0193] On Step **820**, an inverse ansatz parametric circuit may be the applied on the quantum state. In some exemplary embodiments, the parametric quantum circuit may be designed to comprise the inverse ansatz parametric circuit as a sub-circuit thereof.

[0194] In some exemplary embodiments, the inverse ansatz parametric circuit may comprise an inverse of an ansatz parametric circuit, implementing an ansatz. In some exemplary embodiments, the ansatz may comprise a hypothesis or assumption associated with the quantum state (e.g., for guessing the quantum state), that may be based on one or more parameters. In some exemplary embodiments, the ansatz may be assumed to correspond to the quantum state (e.g., under a defined accuracy threshold) when using a certain selection of one or more parameter valuations. In some exemplary embodiments, an inverse of the ansatz may correspond to a subtraction of the ansatz of the quantum state.

[0195] In some exemplary embodiments, the inverse ansatz parametric circuit may comprise one or more quantum gates or other quantum operations that manipulate the set of qubits holding the quantum state of the quantum sensor, e.g., according to the inverse ansatz. In some exemplary embodiments, after the set of qubits is manipulated by the inverse ansatz parametric circuit, the set of qubits may hold one or more manipulated or processed quantum states. In some exemplary embodiments, the inverse ansatz parametric circuit may be configured to apply one or more quantum operations on the qubits that hold the sensor's quantum state, and output a processed state that is based on the sensor's quantum state. For example, the processed state may comprise a subtraction of the ansatz from the initial quantum state.

[0196] On Step **830**, the processed quantum state held by the set of qubits after the inverse ansatz parametric circuit is applied, may or may not be further processed by a detection sub-circuit. In some cases, the parametric quantum circuit may be designed to comprise a detection sub-circuit subsequently to the inverse ansatz parametric circuit. In some cases, the detection sub-circuit may be configured to measure a nullness of one or more states of the set of qubits, such as being measuring a distance between the processed states of the qubits and between a predetermined state, e.g., zero. In some cases, the predetermined state may be zero, and the detection sub-circuit may constitute a zero detection sub-circuit. In other cases, the predetermined state may comprise any other state or value (e.g., 1, 2, 50, or the like), and the detection sub-circuit may be configured to measure a distance between the predetermined state and one or more states of the set of qubits. In some exemplary embodiments, the detection sub-circuit may be configured to indicate whether or not the one or more states of the set of qubits are null, zero, near-zero, or the like.

[0197] In some exemplary embodiments, the quantum computer may be configured to execute the parametric quantum circuit, including the inverse ansatz parametric circuit and the detection sub-circuit, a plurality of times. In some cases, in order to measure the quantum state, the measured phenomenon may be set (e.g., by setting a same atom position) before every measurement of the quantum sensor. In other cases, the measured phenomenon may not be set for subsequent measurements, may be set for a subset of subsequent measurements, or the like.

[0198] In some exemplary embodiments, the plurality of executions may enable to estimate the result of the detection sub-circuit, and infer based thereon whether the valuation of the inverse ansatz parametric circuit was accurate, a level of accuracy thereof, or the like. In some exemplary embodiments, as the number of executions increases, the accuracy of the estimation may increase.

In some cases, instead of measuring the result of the detection sub-circuit, Step **830** may not be implemented, and instead a quantum state resulting from applying the inverse ansatz parametric circuit may be measured directly, compared to a predefined state, or the like.

[0199] On Step **840**, parameters of the inverse ansatz parametric circuit may be adjusted, e.g., based on the execution of the parametric quantum circuit.

[0200] In some exemplary embodiments, the quantum computer may be configured to implement a VQA scheme, in which parameter values of the inverse ansatz parametric circuit are iteratively adjusted, e.g., until the ansatz is determined to be a good enough approximation of the quantum state (such as according to a defined error rate, accuracy score, or any other threshold). In some exemplary embodiments, the physical phenomenon may or may not be configured to be adjusted between iterations of the VQA scheme.

[0201] In some exemplary embodiments, the values of the parameters may be configured to be adjusted between iterations of the VQA scheme, e.g., manually by a user, automatically such as by a search algorithm, or the like. For example, the parameters may be configured to be adjusted between iterations according to a binary search algorithm. In some exemplary embodiments, every iteration of the VQA scheme, a valuation of parameters of the inverse ansatz parametric circuit may be adjusted on Step **840**, and Steps **810**-**840** may be iteratively performed. In some cases, every iteration of the VQA scheme may include a plurality of executions of the parametric quantum circuit, a single execution, or the like.

[0202] In some exemplary embodiments, iterations of the VQA scheme may be executed until valuations of the parameters of the inverse ansatz parametric circuit result with an approximation of a predetermined state such as zero. For example, in case the detection sub-circuit indicates that a result of zero, near zero, or the like was measured, the parameter values that were used for the inverse ansatz parametric circuit may be determined to have generated an inverse of the quantum state. In such cases, the ansatz with the parameter values may be determined to be a correct approximation of the quantum state. For example, the ansatz may comprise an inverse of the inverse ansatz parametric circuit, and the parameter values may be the same parameters that were used for the inverse ansatz parametric circuit, an inverse thereof, or the like.

[0203] In some exemplary embodiments, the detection sub-circuit may indicate a result of zero or near zero in case a distance between one or more measured states of qubits and between a state of zero is less than a predefined threshold. For example, the threshold may represent the maximum allowable deviation or difference between the measured states and the zero state.

[0204] On Step **850**, output data that indicates an approximation of the quantum state may be generated, determined, or the like, based on an output of the parametric quantum circuit. It is noted that data may be considered to “indicate” a state in case the state can be reconstructed, inferred, or determined in any way based on the data. In some cases, the output data may be determined or generated by an output module, based on an output from the detection sub-circuit (e.g., which constitutes the output of the parametric quantum circuit). In some exemplary embodiments, the output data may represent, or indicate, an approximation of the quantum state that was loaded to the parametric quantum circuit from the quantum sensor. In some cases, the output data may comprise data that, when processed, can be used to reconstruct the approximation. In some cases, the output data may not comprise the approximation of the quantum state, but rather may comprise data that can be used to reconstruct the approximation, data that represents the approximation directly or indirectly, or the like. In some exemplary embodiments, the output data may be determined based on the inverse ansatz parametric circuit, the valuations of the set of parameters that was used on the final iteration, or the like.

[0205] In some exemplary embodiments, the output data may comprise a representation of the parametric quantum circuit and the parameter values that were used in the last iteration, measurement, or the like. In some exemplary embodiments, an approximation of the quantum state may be reconstructed by executing the parametric quantum circuit with the same parameter values.

[0206] In some exemplary embodiments, the output data may comprise the ansatz parametric circuit and the parameter values that were used in the last iteration, measurement, or the like, e.g., without the parametric quantum circuit. In some exemplary embodiments, since the inverse ansatz parametric circuit may comprise an inverse of the ansatz parametric circuit, an approximation of the quantum state may be reconstructed by executing the ansatz parametric circuit with the same or inverse parameter values.
[0207] In some exemplary embodiments, the output data may comprise the parameter values without the ansatz parametric circuit, the parametric quantum circuit, or the like. In some exemplary embodiments, an approximation of the quantum state may be reconstructed, at a computing device, by applying the parameter values on a circuit of the computing device. For example, the circuit may correspond to the ansatz parametric circuit, and the computing device may obtain the circuit from a third party, may obtain a representation thereof, may generate the circuit locally, or the like.
[0208] In some exemplary embodiments, the output data may comprise a representation of a value reconstruction circuit, e.g., without the ansatz parametric circuit, the parametric quantum circuit, or the like. In some cases, a value reconstruction circuit may be configured to reconstruct a value associated with the quantum state. For example, the value reconstruction circuit may be configured to reconstruct the parameter values, the approximation of the quantum state, or the like.
[0209] In some exemplary embodiments, the output data may be transmitted to a second quantum computer in a non-quantum communication channel, e.g., via a classical communication channel. For example, the output module may be configured to transmit the output data to a second quantum computer via a classical communication channel. In some exemplary embodiments, the second quantum computer may be part of a quantum cloud computing platform, a quantum computer that is remote and not entangled with the quantum computer, or the like.
[0210] In some exemplary embodiments, the second quantum computer may be enabled to reconstruct the approximation of the quantum state based on the output data. In some exemplary embodiments, the second quantum computer may be enabled to reconstruct the approximation of the quantum state based on the output data, based on an inverse of a value reconstruction circuit, or the like.
[0211] In some exemplary embodiments, the output data may be transmitted to a classical computer. For example, the compression output module may be configured to transmit the output data, e.g., the parameter values. In some exemplary embodiments, the classical computer may be enabled to utilize the output data to approximate the quantum state.
[0212] Referring now to FIG. **9** showing a block diagram of an apparatus, in accordance with some exemplary embodiments of the disclosed subject matter.
[0213] In some exemplary embodiments, Apparatus **900** may comprise one or more Processor(s) **902**. Processor **902** may be a Central Processing Unit (CPU), a microprocessor, an electronic circuit, an Integrated Circuit (IC) or the like. Processor **902** may be utilized to perform computations required by Apparatus **900** or any of its subcomponents. It is noted that Processor **902** may be a traditional processor, and not necessarily a quantum processor.
[0214] In some exemplary embodiments of the disclosed subject matter, Apparatus **900** may comprise an Input/Output (I/O) module **905**. I/O Module **905** may be utilized to provide an output to and receive input from a user, an apparatus, or the like, such as, for example to communicate with quantum hardware, to communicate with a remote quantum computer, to communicate with a classical computer, or the like.
[0215] In some exemplary embodiments, Apparatus **900** may comprise Memory **907**. Memory **907** may be a hard disk drive, a Flash disk, a Random Access Memory (RAM), a memory chip, or the like. In some exemplary embodiments, Memory **907** may retain program code operative to cause Processor **902** to perform acts associated with any of the subcomponents of Apparatus **900**. Memory **907** may comprise one or more components as detailed below, implemented as

executables, libraries, static libraries, functions, or any other executable components.
[0216] In some exemplary embodiments, Memory **907** may comprise a Data Loader **910**. Data Loader **910** may be configured to load at least one quantum state from a quantum sensor on one or more qubits.
[0217] In some exemplary embodiments, Memory **907** may comprise an Ansatz **920**, which may be configured to attempt to approximate the quantum state. For example, Ansatz **920** may comprise a sub-circuit that manipulates qubits according to a determined ansatz and associated parameter values, under the assumption that a valuation of the parameters exists such that the sub-circuit approximated the quantum state.
[0218] In some exemplary embodiments, Memory **907** may comprise a Detector **930**, which may be configured to obtain qubits that are manipulated by an inverse of Ansatz **920**, and detect whether they correspond to a predefined state such as ground state.
[0219] In some exemplary embodiments, Memory **907** may comprise a Circuit Executer **940**, which may be configured to execute a parametric quantum circuit that is initialized by Data Loader **910** and includes an inverse of Ansatz **920**, Detector **930**, or the like. For example, Circuit Executer **940** may execute the parametric quantum circuit on Quantum Execution Platform **990** a plurality of time for each valuation of parameters, a single time for each valuation of parameters, or the like. In some exemplary embodiments, Quantum Execution Platform **990** may comprise at least one quantum computer, at least one quantum computing cloud, a combination thereof, or the like.
[0220] In some exemplary embodiments, Memory **907** may comprise a Parameter Adjuster **950**, which may be configured to obtain an outcome of an execution by Circuit Executer **940**, and based thereon determine whether to adjust parameters of the inverse of Ansatz **920**, how to adjust the parameters, or the like. For example, in case Detector **930** indicates that an inverse of Ansatz **920** did not result with a zero or near zero state, Parameter Adjuster **950** may adjust parameters of Ansatz **920** to values that are estimated to result with a state that is nearer to ground state. Otherwise, in case an inverse of Ansatz **920** did result with a zero or near zero state, Parameter Adjuster **950** may indicate to Output Module **960** that a satisfying approximation of the quantum state was found.
[0221] In some exemplary embodiments, Memory **907** may comprise an Output Module **960**, which may be configured to generate output data, e.g., a compressed version of an approximation of the quantum state. For example, Output Module **960** may generate output data to include parameters of Ansatz **920**, a representation of Ansatz **920**, or the like. In some cases, Output Module **960** may communicate the output data, via a classical medium such as I/O Module **905**, to one or more classical computers, remote quantum computers, or the like.
[0222] One technical problem dealt with by the disclosed subject matter is setting one or more states to qubits with unknown states, mid-program. For example, it may be desired to set a qubit with an unknown state to a target state, and continue execution thereafter with the modified state.
[0223] In some exemplary embodiments, according to a first technical solution, a qubit with an unknown state may be set to a target state by resetting the unknown state of the qubit to zero or ground state, and then adding the target state to the ground state using a new sub-circuit. For example, the unknown state of the qubit may be reset by measuring the qubit, such as via sampling of a probabilistic result. When sampling is performed, the distribution may collapse, which may reset the state of the qubit to ground state. In other cases, the qubit may be reset in any other way. In some exemplary embodiments, the target state may be added to the qubit, such as by adding a subcircuit that is designed to add the target state to an input qubit to which it is applied.
[0224] In some exemplary embodiments, the first technical solution may have one or more drawbacks. For example, in some cases, resetting the state of the qubit may damage quantum properties of other qubits in the remaining circuit, such as entanglement properties of qubits that are entangled with the qubit. As another example, the first technical solution may not be efficient, such as in case the target state has a value that is similar or in close proximity to the unknown value

of the qubit. According to this example, resetting the qubit to ground state may be inefficient, resource consuming, or the like, since using the current unknown state of the qubit for the construction of the target state may be more efficient and utilize less resources. For example, in case the current unknown state of the qubit corresponds to a value of 2.8736, and the target state for the qubit corresponds to a value of 4.8736, reconstructing the 4.8736 from ground state may be inefficient, resource consuming, or the like, while adding a value of two may be efficient and resource conserving.

[0225] In some exemplary embodiments, a second technical solution may be configured to overcome the drawbacks of the first technical solution. In some exemplary embodiments, according to the second technical solution, a qubit with an unknown state may be set to a target state by performing tomography measurements to calculated the current unknown state of the qubit, and adding a sub-circuit that changes the current state of the qubit to the target state. For example, in case the tomography measurements indicate that the current state of the qubit is 2.8736, and the target state corresponds to a value of 4.8736, a simple sub-circuit that adds a value of 2 may be added to the circuit, and may be applied to the qubit.

[0226] In some exemplary embodiments, although this technical solution may be advantageous for scenarios in which the target state has a value that is similar or in close proximity to the unknown value of the qubit, the second technical solution may have one or more drawbacks. For example, tomography measurements may be resource consuming, expensive in computational resource, require a large number of executions, time consuming, or the like.

[0227] In some exemplary embodiments, it may be desired to overcome the drawbacks of the first and second technical solutions, and provide a solution that is more efficient than both first and second technical solutions in most scenarios.

[0228] Another technical problem dealt with by the disclosed subject matter is providing a quantum debugger that is capable of setting one or more states to qubits with unknown states, mid-program. For example, it may be desired to provide a quantum debugger, such as a quantum debugger corresponding to FIGS. **1**A-**1**F and FIG. **2**, that has versatile abilities, such as setting values to variables mid-program and continuing execution thereafter with the modified value.

[0229] One technical solution provided by the disclosed subject matter is using a VQA framework for setting a qubit with an unknown state to a target state. For example, the VQA framework may at least partially correspond to the VQA scheme of FIGS. **7**-**8**.

[0230] In some exemplary embodiments, an original quantum circuit may be obtained, designed, received from a third party, or the like. In some exemplary embodiments, the original quantum circuit may comprise a qubit, also referred to as the inspected qubit, with an unknown state. For example, the state of the qubit may be unknown in one or more cycles of the original quantum circuit.

[0231] In some exemplary embodiments, it may be desired to set a target state to the qubit at a specified cycle or set of cycles, in which the state of the qubit is unknown. For example, the state of the qubit may be fully unknown, unknown in part, or the like. In some cases, it may be desired to set the target state to the qubit after one or more operations are applied to the qubit or to other qubits, such as regardless of any specified cycle. In some exemplary embodiments, a ‘target cycle’ may refer to one or more cycles that are specified for changing the state of the qubit, or to one or more cycles that are subsequent to the one or more operations.

[0232] In some exemplary embodiments, in order to set the target state to the qubit at a target cycle, a VQA framework may be applied in order to obtain a sub-circuit (e.g., a ‘target sub-circuit’) that transfers the unknown state of the qubit to the target state, and the sub-circuit may be applied to the qubit in the original quantum circuit at the target cycle, thereby generating a modified quantum circuit that changed the value of the qubit mid-program. In some exemplary embodiments, the sub-circuit may be determined using the VQA framework, without performing resource-consuming tomography measurements, and without resetting the quantum state of the qubit.

[0233] In some cases, before implementing the VQA framework, the unknown original state of the qubit may be measured using simple measurements (without using tomography measurements) to ensure that it does not correspond to the target state. According to this example, after the simple measurements are conducted, the state of the qubit may remain unknown (except for the fact that it does not correspond to the target state), and the need for implementing the VQA framework may be determined. In other cases, the VQA framework may be implemented without performing preliminary measurements.
[0234] In some exemplary embodiments, the VQA framework may involve an iterative process where a parametric quantum circuit is constructed, executed, and measured; the measurement results may determine the construction of the next iterated circuit. In some exemplary embodiments, the VQA iterations may be executed for different logical variations of logical parameters associated with the parametric quantum circuit, resulting from different valuations of the logical parameters.
[0235] In some exemplary embodiments, the parametric quantum circuit may be constructed to comprise an ansatz parametric circuit and a detection sub-circuit. For example, the ansatz parametric circuit may be scheduled to be executed before the detection sub-circuit. In some exemplary embodiments, the ansatz parametric circuit may represent, or implement, an ansatz (e.g., a hypothesis, or educated guess) associated with a difference between the unknown quantum state of the qubit and a target state. For example, the hypothesis or ansatz associated to the ansatz parametric circuit may be that there exists a valuation of the logical parameters of the ansatz parametric circuit, that, when used to construct the ansatz parametric circuit, will cause the ansatz parametric circuit to produce the difference between the unknown quantum state and the target state (under a certain deviation threshold).
[0236] In some exemplary embodiments, the ansatz parametric circuit implementing the ansatz may comprise a quantum circuit (or subcircuit) that, when executed on a quantum execution platform, is configured to output a quantum state, in accordance with the ansatz. In some exemplary embodiments, the ansatz parametric circuit may be designed as a hardware-oriented ansatz, a physics-oriented ansatz, or as any other type of ansatz. In some exemplary embodiments, a hardware-oriented ansatz may be formulated as a random and maximally condensed quantum circuit that can be efficiently executed on a quantum computer, e.g., in terms of cycle-wise depth, error rates, costs, gate count, or the like. In some exemplary embodiments, any other types of ansätze and associated parameters may be used.
[0237] In some exemplary embodiments, during each VQA iteration, the unknown quantum state of the qubit may be fed to the ansatz parametric circuit, and an output from the ansatz parametric circuit may be provided to the detection sub-circuit of the parametric quantum circuit. In some exemplary embodiments, the detection sub-circuit may be configured to determine whether the output from the ansatz parametric circuit corresponds to the target state. For example, the detection sub-circuit may be constructing in accordance with the target state, and may differ for different target states.
[0238] In some exemplary embodiments, for each valuation of the set of parameters, a quantum computer may be configured to execute a resulting parametric quantum circuit. For example, the parametric quantum circuit may include the adjusted ansatz parametric circuit, one or more predefined variations of the detection sub-circuit, or the like. In some exemplary embodiments, during a VQA iteration, the parametric quantum circuit may be executed a plurality of times, once, or the like. In some exemplary embodiments, the quantum computer may be configured to execute the parametric quantum circuit a plurality of times, such as in order to accurately measure the output quantum state.
[0239] For example, each VQA iteration, the parametric quantum circuit may be executed a plurality of times and the detection sub-circuit may determine, in each execution, a probability that the output from the ansatz parametric circuit corresponds to the target state. According to this

example, based on the plurality of executions, a statistically significant result from the detection sub-circuit may be determined, and used for adjusting the logical parameters of the parametric quantum circuit.

[0240] In some exemplary embodiments, a valuation of the set of parameters may be determined by a classical processor, such as based on heuristics, a predictor, a search algorithm, or the like. In some exemplary embodiments, the classical processor may obtain execution results associated with a VQA iteration, and adjust the logical parameters of the parametric quantum circuit accordingly. For example, each execution result may comprise a corresponding output from the detection sub-circuit, and the classical processor may estimate a statistically significant conclusion, based on the plurality of execution results, indicating whether or not the outputs from the ansatz parametric circuit corresponds to the target state.

[0241] In some exemplary embodiments, measurements of the executions' outputs may be used to update the parameters of the ansatz parametric circuit iteratively, until the algorithm converges to a solution that minimizes an objective function associated with the optimization problem. For example, the objective function may be configured to measure a difference between an output from the parametric quantum circuit and a target state for the qubit, and the optimization function may be configured to minimize this difference to zero, near zero, or the like.

[0242] In some exemplary embodiments, the VQA iterations may terminate, when the detection sub-circuit indicates that the output from the ansatz parametric circuit corresponds to the target state. For example, the output from the ansatz parametric circuit may be determined to correspond to the target state in case an estimated distance between the output and the target state falls within a defined acceptable range, ensuring a level of accuracy deemed satisfactory. In some cases, such acceptable range may be defined by a user, defined by default settings, heuristics, or the like. In other cases, the output from the ansatz parametric circuit may be determined to correspond to the target state based on any other parameter, e.g., statistical properties of the gathered information, the standard deviation of the measurements, estimated confidence in the result, or the like.

[0243] In some exemplary embodiments, at termination of the iterative process of the VQA, the original quantum circuit may be modified, revised, adjusted, or the like, to include therein a sub-circuit that corresponds to the ansatz parametric circuit. For example, the ansatz parametric circuit may be constructed and applied to the qubit, at the target cycle, thereby changing its state to the target state without measuring at any point the unknown original state of the qubit and without resetting the unknown original state of the qubit to ground state.

[0244] One technical effect obtained by the disclosed subject matter is providing a technique for setting a qubit with an unknown state to a target state, without requiring resource-consuming tomography measurements and without resetting the unknown original state of the qubit to ground state. In some exemplary embodiments, the disclosed subject matter utilizes an ansatz parametric circuit as a transformer that moves the quantum state of the qubit from the original unknown state to the target state, as part of a VQA framework, without necessarily measuring or determining the original unknown state at any point in time. This enables to reduce computational resources, error rates, execution time, or the like.

[0245] Another technical effect obtained by the disclosed subject matter is enabling to set a target state on a qubit with an unknown state, in a manner that may be more efficient than the first and second technical solutions described above. In some exemplary embodiments, the disclosed subject matter relies on a VQA framework that includes an ansatz parametric circuit and a detection sub-circuit. In some exemplary embodiments, constructing and utilizing the detection sub-circuit may be resource conserving and comprise less calculations compared to tomography measurements, thereby reducing a number of resources and a cost of setting a qubit to a target state.

[0246] Yet another technical effect obtained by the disclosed subject matter is enabling to set target states on a plurality of qubits. For example, since the disclosed subject matter provides a resource conserving solution to the problem of setting a target state on a qubit with an unknown state, a

same set of resources that was previously sufficient for setting a single qubit or qubit register to a target state, may now suffice for setting a plurality of target states on a respective plurality of qubits.

[0247] Yet another technical effect of utilizing the disclosed subject matter is to provide a solution enabling to debug quantum programs, using a reduced number of resources.

[0248] Referring now to FIG. **10**, showing a schematic block diagram, in accordance with some exemplary embodiments of the disclosed subject matter.

[0249] In some exemplary embodiments, Block Diagram **1000** may depict blocks, each of which represents a component, stage, or subsystem of the disclosed subject matter, while interconnections between the blocks may illustrate how these components interact or are related in terms of functionality or information flow.

[0250] As depicted in FIG. **10**, a flow of Block Diagram **1000** starts with loading the unknown quantum state of a qubit (a single qubit or a qubit register) to a set of one or more qubits of Quantum Circuit **1004**. For example, Unknown Quantum State **1041** may represent the set of qubits holding the unknown quantum state. In some exemplary embodiments, Unknown Quantum State **1041** may be initialized with the unknown quantum state at a start of Quantum Circuit **1004**, which are subsequently manipulated by Ansatz **1043** and State Detection **1045**. In some exemplary embodiments, Unknown Quantum State **1041** may be absent of any logical gates.

[0251] In some exemplary embodiments, when executing Quantum Circuit **1004**, Unknown Quantum State **1041** may be manipulated by Ansatz **1043**. In some exemplary embodiments, Ansatz **1043** may implement an ansatz attempting to approximate the difference between the unknown quantum state of the inspected qubit and the target state, and to add this difference to the unknown quantum state. In some exemplary embodiments, Ansatz **1043** may apply one or more quantum operations to Unknown Quantum State **1041**, and generate a processed version of the unknown quantum state, which may be fed to State Detection **1045**. In some exemplary embodiments, State Detection **1045** may be designed to determine whether the processed data from Ansatz **1043** is a predefined quantum state, e.g., the target state. For example, State Detection **1045** may output an indication of the level of similarity, or distance, between the processed data and the target state.

[0252] In some exemplary embodiments, Quantum Circuit **1004** may be executed by Execute **1006**, and its results may be measured by Measure **1008**. For example, Execute **1006** may execute Quantum Circuit **1004** once, a plurality of times, or the like, while using the same logical parameters of Ansatz **1043**. In some cases, Measure **1008** may measure an output state of each execution of Quantum Circuit **1004**, which may constitute an indication provided by State Detection **1045**.

[0253] In some exemplary embodiments, a VQA scheme may be implemented to iteratively adjust parameter values of Ansatz **1043**, according to outputs from State Detection **1045**. For example, in case the indication from State Detection **1045** indicates that the output from Ansatz **1043** is greater than the target state, parameter values of Ansatz **1043** may be adjusted to reduce a value of the state produced by Ansatz **1043**. As another example, in case the indication from State Detection **1045** indicates that the output from Ansatz **1043** is lesser than the target state, parameter values of Ansatz **1043** may be adjusted to increase the value of the state produced by Ansatz **1043**. In other cases, the valuations for the set of parameters may be selected by a classical optimization based on any other algorithm, scheme, or the like, e.g., a greedy search algorithm, a non-greedy search algorithm, manually by a user, a binary search, or the like.

[0254] In some exemplary embodiments, the parameters of Ansatz **1043** may be adjusted as part of the VQA scheme, thereby providing different valuations to the parameters every VQA iteration, and creating respective different variations of Ansatz **1043** for different VQA iterations. It is noted that a single VQA iteration may comprise a plurality of executions of Quantum Circuit **1004**, such as in order to increase the accuracy of the output, in order to increase the confidence of the

indication from State Detection **1045**, or the like.
[0255] In some exemplary embodiments, the output from State Detection **1045** may be used, by a classical processor, to update the parameters of Ansatz **1043** iteratively, until State Detection **1045** detects a state that corresponds to the target state. In such cases, the modified Ansatz **1043** that corresponds to the updated parameters may produce a state that corresponds to the difference between the unknown quantum state and the target state.
[0256] In some exemplary embodiments, in case State Detection **1045** detects a state that corresponds to the target state, Output Data **1010** may be generated. For example, Output Data **1010** may be generated to comprise Ansatz **1043**, a reconstruction of Ansatz **1043**, the parameter values of Ansatz **1043** that were used in the last VQA iteration, or the like.
[0257] In some cases, the target state may comprise a mixed quantum state, which, according to the Schrödinger-HJW theorem, may represent a statistical mixture of multiple pure states. In such cases, a pure state may be prepared on the qubit along with one or more additional auxiliary qubits, qubit registers, or the like, such that the state on the qubit alone corresponds to the target state. The number of auxiliary qubits may be identical or smaller than the number of qubits of the inspected qubit. According to this scenario, Unknown Quantum State **1041** may obtain the states of both the qubit and the auxiliary qubits, while State Detection **1045** may be designed to measure the distance between the target state and the inspected qubit alone (regardless of the state of the auxiliary qubits).
[0258] For example, in case the inspected qubit is denoted by A, and the auxiliary qubits are denoted by B, a pure state may be prepared on a joint register that includes both A and B. The state on A alone may be referred to as the ‘reduced state’ of A, and this state may be replaced with the target state as part of a purification process. In some exemplary embodiments, the VQA framework of Block Diagram **1000** may obtain states of A and B, and Ansatz **1043** may transform both the states of A and B to a state in which the reduced state of A, after being processed by Ansatz **1043** (e.g., a purified state on the joint A and B), results with the target state. After the execution, the qubits in B may be in a dirty quantum state.
[0259] Referring now to FIG. **11**, showing an exemplary flowchart diagram of a method, in accordance with some exemplary embodiments of the disclosed subject matter.
[0260] On Step **1110**, at least one qubit may be loaded with an unknown quantum state at an initial cycle of a parametric quantum circuit.
[0261] On Step **1120**, an ansatz parametric circuit may be configured to receive the unknown quantum state from the at least one qubit, and be applied thereon. In some exemplary embodiments, the ansatz parametric circuit may be configured to manipulate the unknown quantum state, and output a processed quantum state based thereon.
[0262] On Step **1130**, a detection sub-circuit may be configured to obtain the processed quantum state from the ansatz parametric circuit, and measure a distance between the processed quantum state and a target state. In some exemplary embodiments, the target state may comprise a predetermined quantum state, which may or may not be obtained from a user. In some exemplary embodiments, the detection sub-circuit may indicate, as an output, a distance measure of the processed quantum state from the target state.
[0263] On Step **1140**, a quantum computer may be configured to execute the parametric quantum circuit a plurality of times. In some exemplary embodiments, the quantum computer may be configured to implement a VQA scheme to iteratively set parameter values of the ansatz parametric circuit until obtaining a parameter value that causes the parametric quantum circuit to output an approximation of zero as the distance measure. In some exemplary embodiments, the distance measure of zero may indicate that the processed quantum state from the ansatz parametric circuit corresponds to the target state.
[0264] On Step **1150**, an output module may be configured to provide output data that corresponds to the ansatz parametric circuit. For example, the output data may indicate the parameter value. As

another example, the output data may comprise a representation of the ansatz parametric circuit. As another example, the output data may comprise a quantum circuit that reconstructs the ansatz parametric circuit according to the parameter value.

[0265] In some exemplary embodiments, an original quantum circuit may comprise a qubit with the unknown quantum state. In some exemplary embodiments, a representation of the original quantum circuit may be configured to be modified based on the output data. In some exemplary embodiments, the representation may be modified by applying the ansatz parametric circuit to the qubit with the unknown quantum state during an intermediate cycle of an execution of the original quantum circuit.

[0266] The present disclosed subject matter may be a system, a method, and/or a computer program product. The computer program product may include a computer readable storage medium (or media) having computer readable program instructions thereon for causing a processor to carry out aspects of the present disclosed subject matter.

[0267] The computer readable storage medium can be a tangible device that can retain and store instructions for use by an instruction execution device. The computer readable storage medium may be, for example, but is not limited to, an electronic storage device, a magnetic storage device, an optical storage device, an electromagnetic storage device, a semiconductor storage device, or any suitable combination of the foregoing. A non-exhaustive list of more specific examples of the computer readable storage medium includes the following: a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), a static random access memory (SRAM), a portable compact disc read-only memory (CD-ROM), a digital versatile disk (DVD), a memory stick, a floppy disk, a mechanically encoded device such as punch-cards or raised structures in a groove having instructions recorded thereon, and any suitable combination of the foregoing. A computer readable storage medium, as used herein, is not to be construed as being transitory signals per se, such as radio waves or other freely propagating electromagnetic waves, electromagnetic waves propagating through a waveguide or other transmission media (e.g., light pulses passing through a fiber-optic cable), electrical signals transmitted through a wire, Quantum Random Access Memory (QRAM), photons, trapped ions, lasers, cold atoms, or the like.

[0268] Computer readable program instructions described herein can be downloaded to respective computing/processing devices from a computer readable storage medium or to an external computer or external storage device via a network, for example, the Internet, a local area network, a wide area network and/or a wireless network. The network may comprise copper transmission cables, optical transmission fibers, wireless transmission, routers, firewalls, switches, gateway computers and/or edge servers. A network adapter card or network interface in each computing/processing device receives computer readable program instructions from the network and forwards the computer readable program instructions for storage in a computer readable storage medium within the respective computing/processing device.

[0269] Computer readable program instructions for carrying out operations of the present disclosed subject matter may be assembler instructions, instruction-set-architecture (ISA) instructions, machine instructions, machine dependent instructions, microcode, firmware instructions, state-setting data, or either source code or object code written in any combination of one or more programming languages, including an object oriented programming language such as Smalltalk, C++ or the like, and conventional procedural programming languages, such as the “C” programming language or similar programming languages. The computer readable program instructions may execute entirely on the user's computer, partly on the user's computer, as a stand-alone software package, partly on the user's computer and partly on a remote computer or entirely on the remote computer or server (or a group of multiple remote servers). In the latter scenario, the remote computer may be connected to the user's computer through any type of network, including a local area network (LAN) or a wide area network (WAN), or the connection may be made to an

external computer (for example, through the Internet using an Internet Service Provider). In some embodiments, electronic circuitry including, for example, programmable logic circuitry, field-programmable gate arrays (FPGA), or programmable logic arrays (PLA) may execute the computer readable program instructions by utilizing state information of the computer readable program instructions to personalize the electronic circuitry, in order to perform aspects of the present disclosed subject matter.

[0270] Aspects of the present disclosed subject matter are described herein with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems), and computer program products according to embodiments of the disclosed subject matter. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by computer readable program instructions.

[0271] These computer readable program instructions may be provided to a processor of a general-purpose computer, special purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks. These computer readable program instructions may also be stored in a computer readable storage medium that can direct a computer, a programmable data processing apparatus, and/or other devices to function in a particular manner, such that the computer readable storage medium having instructions stored therein comprises an article of manufacture including instructions which implement aspects of the function/act specified in the flowchart and/or block diagram block or blocks.

[0272] The computer readable program instructions may also be loaded onto a computer, other programmable data processing apparatus, or other device to cause a series of operational steps to be performed on the computer, other programmable apparatus or other device to produce a computer implemented process, such that the instructions which execute on the computer, other programmable apparatus, or other device implement the functions/acts specified in the flowchart and/or block diagram block or blocks.

[0273] The flowchart and block diagrams in the Figures illustrate the architecture, functionality, and operation of possible implementations of systems, methods, and computer program products according to various embodiments of the present disclosed subject matter. In this regard, each block in the flowchart or block diagrams may represent a module, segment, or portion of instructions, which comprises one or more executable instructions for implementing the specified logical function(s). In some alternative implementations, the functions noted in the block may occur out of the order noted in the figures. For example, two blocks shown in succession may, in fact, be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved. It will also be noted that each block of the block diagrams and/or flowchart illustration, and combinations of blocks in the block diagrams and/or flowchart illustration, can be implemented by special purpose hardware-based systems that perform the specified functions or acts or carry out combinations of special purpose hardware and computer instructions.

[0274] The terminology used herein is for the purpose of describing particular embodiments only and is not intended to be limiting of the disclosed subject matter. As used herein, the singular forms “a”, “an” and “the” are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the terms “comprises” and/or “comprising,” when used in this specification, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof.

[0275] The corresponding structures, materials, acts, and equivalents of all means or step plus function elements in the claims below are intended to include any structure, material, or act for

performing the function in combination with other claimed elements as specifically claimed. The description of the present disclosed subject matter has been presented for purposes of illustration and description but is not intended to be exhaustive or limited to the disclosed subject matter in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art without departing from the scope and spirit of the disclosed subject matter. The embodiment was chosen and described in order to best explain the principles of the disclosed subject matter and the practical application, and to enable others of ordinary skill in the art to understand the disclosed subject matter for various embodiments with various modifications as are suited to the particular use contemplated.

Claims

1. A system comprising: a quantum computer that is configured to execute a parametric quantum circuit a plurality of times, wherein the parametric quantum circuit comprises: at least one qubit with an unknown quantum state at an initial cycle of the parametric quantum circuit, an ansatz parametric circuit, the ansatz parametric circuit is configured to receive the unknown quantum state from the at least one qubit, whereby the ansatz parametric circuit is configured to output a processed quantum state, and a detection sub-circuit, the detection sub-circuit is configured to indicate a distance measure of the processed quantum state from a target state, wherein said quantum computer is configured to implement a Variational Quantum Algorithm (VQA) scheme to iteratively set parameter values of the ansatz parametric circuit until obtaining a parameter value that causes the parametric quantum circuit to output an approximation of zero as the distance measure; and an output module, said output module is configured to provide output data that corresponds to the ansatz parametric circuit.
2. The system of claim 1, wherein system further comprises an original quantum circuit, the original quantum circuit comprising a qubit with the unknown quantum state, wherein a representation of the original quantum circuit is configured to be modified based on the output data.
3. The system of claim 2, wherein said modifying the representation comprises applying the ansatz parametric circuit to the qubit with the unknown quantum state during an intermediate cycle of an execution of the original quantum circuit.
4. The system of claim 1, wherein the target state is a predetermined quantum state.
5. The system of claim 1, wherein the target state is obtained from a user.
6. The system of claim 1, wherein the output data indicates the parameter value.
7. The system of claim 1, wherein the output data comprises a representation of the ansatz parametric circuit.
8. The system of claim 1, wherein the output data comprises a quantum circuit that reconstructs the ansatz parametric circuit according to the parameter value.
9. A method comprising: obtaining a target state; obtaining a representation of an original quantum circuit, the original quantum circuit comprising a qubit with an unknown quantum state; generating a parametric quantum circuit, said generating comprises: setting the unknown quantum state on at least one qubit at an initial cycle of the parametric quantum circuit, generating an ansatz parametric circuit to receive the unknown quantum state from the at least one qubit, and to output a processed quantum state, generating a detection sub-circuit to indicate a distance measure of the processed quantum state from the target state, executing the parametric quantum circuit a plurality of times on a quantum computer, based on said executing, iteratively adjusting parameter values of the ansatz parametric circuit until obtaining a parameter value that, when executed by said executing, causes the parametric quantum circuit to output an approximation of zero as the distance measure, wherein said iteratively adjusting is part of a Variational Quantum Algorithm (VQA) scheme, and upon determining that the distance measure corresponds to the approximation of zero, returning output data corresponding to the ansatz parametric circuit; and adjusting the representation of the original

quantum circuit based on the output data.

10. The method of claim 9, wherein said adjusting the representation comprises applying the ansatz parametric circuit to the qubit with the unknown quantum state during an intermediate cycle of an execution of the original quantum circuit.

11. The method of claim 9, wherein the target state is a predetermined quantum state.

12. The method of claim 9, wherein the target state is obtained from a user.

13. The method of claim 9, wherein the output data indicates the parameter value.

14. The method of claim 9, wherein the output data comprises a representation of the ansatz parametric circuit.

15. The method of claim 9, wherein the output data comprises a quantum circuit that reconstructs the ansatz parametric circuit according to the parameter value.

16. An apparatus comprising a processor and coupled memory, said processor being adapted to perform: obtaining a target state; obtaining a representation of an original quantum circuit, the original quantum circuit comprising a qubit with an unknown quantum state; generating a parametric quantum circuit, said generating comprises: setting the unknown quantum state on at least one qubit at an initial cycle of the parametric quantum circuit, generating an ansatz parametric circuit to receive the unknown quantum state from the at least one qubit, and to output a processed quantum state, generating a detection sub-circuit to indicate a distance measure of the processed quantum state from the target state, executing the parametric quantum circuit a plurality of times on a quantum computer, based on said executing, iteratively adjusting parameter values of the ansatz parametric circuit until obtaining a parameter value that, when executed by said executing, causes the parametric quantum circuit to output an approximation of zero as the distance measure, wherein said iteratively adjusting is part of a Variational Quantum Algorithm (VQA) scheme, and upon determining that the distance measure corresponds to the approximation of zero, returning output data corresponding to the ansatz parametric circuit; and adjusting the representation of the original quantum circuit based on the output data.

17. The apparatus of claim 16, wherein said adjusting the representation comprises applying the ansatz parametric circuit to the qubit with the unknown quantum state during an intermediate cycle of an execution of the original quantum circuit.

18. The apparatus of claim 16, wherein the output data indicates the parameter value.

19. The apparatus of claim 16, wherein the output data comprises a representation of the ansatz parametric circuit.

20. The apparatus of claim 16, wherein the output data comprises a quantum circuit that reconstructs the ansatz parametric circuit according to the parameter value.