Hardware requirements for trapped-ion-based verifiable blind quantum computing with a measurement-only client

van Dam, J.; Avis, G.; Propp, T.B.; Horta Ferreira da Silva, F.; Slater, J.A.; Northup, T.E.; Wehner, S.D.C.

DOI
10.1088/2058-9565/ad6eb2

Publication date
2024

Document Version
Final published version

Published in
Quantum Science and Technology

Hardware requirements for trapped-ion-based verifiable blind quantum computing with a measurement-only client

IOP Publishing Quantum Sci. Technol. 9 (2024) 045031 https://doi.org/10.1088/2058-9565/ad6eb2

Quantum Science and Technology

OPEN ACCESS

RECEIVED
30 April 2024

REVISED
5 August 2024

ACCEPTED FOR PUBLICATION
13 August 2024

PUBLISHED
27 August 2024

PAPER

Hardware requirements for trapped-ion-based verifiable blind quantum computing with a measurement-only client

J van Dam[1,2,3,*], **G Avis**[1,2,3,4], **Tz B Propp**[1,2,3], **F Ferreira da Silva**[1,2,3], **J A Slater**[5], **T E Northup**[6] and **S Wehner**[1,2,3]

[1] QuTech, Delft University of Technology, Lorentzweg 1, 2628 CJ Delft, The Netherlands
[2] Kavli Institute of Nanoscience, Delft University of Technology, Lorentzweg 1, 2628 CJ Delft, The Netherlands
[3] Quantum Computer Science, EEMCS, Delft University of Technology, Lorentzweg 1, 2628 CJ Delft, The Netherlands
[4] College of Information and Computer Science, University of Massachusetts, 140 Governors Dr, Amherst, MA 01002, United States of America
[5] Q*Bird, Delftechpark 1, 2628 XJ Delft, The Netherlands
[6] Institut für Experimentalphysik, Universität Innsbruck, Technikerstraße 25, 6020 Innsbruck, Austria
[*] Author to whom any correspondence should be addressed.

E-mail: j.vandam-3@tudelft.nl

Keywords: blind quantum computing, trapped-ions, hardware requirements

Abstract

In blind quantum computing (BQC), a user with a simple client device can perform a quantum computation on a remote quantum server such that the server cannot gain knowledge about the computation. Here, we numerically investigate hardware requirements for verifiable BQC using an ion trap as server and a distant measurement-only client. While the client has no direct access to quantum-computing resources, it can remotely execute quantum programs on the server by measuring photons emitted by the trapped ion. We introduce a numerical model for trapped-ion quantum devices in NetSquid, a discrete-event simulator for quantum networks. Using this, we determine the minimal hardware requirements on a per-parameter basis to perform the verifiable BQC protocol. We benchmark these for a five-qubit linear graph state, with which any single-qubit rotation can be performed, where client and server are separated by 50 km. Current state-of-the-art ion traps satisfy the minimal requirements on a per-parameter basis, but all current imperfections combined make it impossible to perform the blind computation securely over 50 km using existing technology. Using a genetic algorithm, we determine the set of hardware parameters that minimises the total improvements required, finding directions along which to improve hardware to reach our threshold error probability that would enable experimental demonstration. In this way, we lay a path for the near-term experimental progress required to realise the implementation of verifiable BQC over a 50 km distance.

1. Introduction

Quantum computers may outperform classical computers in a variety of tasks [1–3], but these advantages are so far inaccessible. Moreover, despite progress in realising these devices across a variety of physical platforms [4–8], building and running quantum computers is associated with a large financial cost [9–11]. Cloud-based access to quantum servers eliminates the need for users to own large and expensive devices themselves [12] but is unsuitable for use cases involving sensitive data, in which a user requires access to the computational power of the quantum server without revealing the input data, the computation or the output to the owner of the server [13–15].

Blind quantum computing (BQC) is a technique with which a client can execute quantum algorithms at a remote quantum server without the input, the computation, or its outcome being revealed (apart from an upper bound on the size of the computation) [16]. Preferably, the client is realised as cheaply as possible, to

help make quantum computing more widely available. Two options for realising such a client are for it to have the ability to either send single photons [16] or measure them [17, 18]. Alternatively, one can make use of a single-qubit-gate-performing client [19] or a multi-server approach, in combination with a completely classical client [20–23]. In this work, we assume a single-server setup where the quantum capabilities of the client are limited to making measurements.

The initial BQC protocol, based on measurement-based quantum computation [24, 25], was later expanded to include verification to test for correctness; approaches for verification are summarised in [26]. In such verifiable BQC (VBQC) protocols, the client can abort if the outputs of certain tests (either trap based [27], stabiliser based [28] or classical but introducing computational assumptions [29]) are not as expected.

In realistic near-term noisy quantum devices [30], imperfections are inevitable. In verifying tests, imperfections and noise in the system can be mistaken for malicious behaviour of the server, resulting in a computation that will be aborted constantly and the client gaining little information about the operations of the server. There are ways to deal with noise in non-verified protocols [31], and later a noise-robust verified BQC (rVBQC) protocol was introduced aswell [32]. This rVBQC protocol tolerates imperfections from noise or malicious behaviour provided that the server does not fail more than 25% of verifying tests employed by the client. The robustness to noise is realised by repeating the computation multiple times and performing classical error correction (majority voting) on the results. Already, there have been proof-of-principle demonstrations of BQC [33, 34] and rVBQC [35] in laboratory settings. For real-world practicality, however, the client needs to be able to be spatially remote from the server, which will require improvements to existing quantum computing and communication hardware.

In this paper, we determine the requirements for performing rVBQC at a metropolitan scale (i.e. the scale of a large city) of 50 km using a trapped-ion-based server and a measurement-only client as depicted in figure 1, using the 25% error tolerance as a threshold. In measurement-based quantum computing, any single-qubit gate can be performed using a five-qubit linear graph state (i.e. five qubits, each in a superposition state, with controlled-Z operations between them) [36]. We use this five-qubit graph state to benchmark the performance of the protocol. We focus on rVBQC because this is feasibly achievable in the near-term.

We investigate this implementation of the rVQBC protocol numerically using NetSquid, a discrete event simulator for quantum networks [37]. To this end, we introduce a framework for the modelling of trapped-ion quantum servers, including a NetSquid library [38], along with a model of a measurement-only client. The simulation is hardware motivated and takes a set of hardware parameters as input. Using this, we

1. Identify the per-parameter minimal requirements for hardware to allow for rVBQC. In each case, we assume perfect performance of all other parameters apart from fibre attenuation. This gives us a strict lower bound for each parameter, which is compared to state-of-the-art performance. We show that current state-of-the-art ion traps satisfy absolute minimal requirements on a per-parameter basis, but all current imperfections combined make it impossible to perform rVBQC securely over 50 km using existing technology. These results are summarised in figure 4;
2. Identify the set of hardware parameters that minimises the total improvement needed over current state-of-the-art parameters to allow for a successful implementation of rVBQC. This reveals which parameters need the most improvement and how far we need to improve them to enable a metropolitan-scale application of rVBQC. We do this by combining our requirement on the error probability with the cost [39, 40] of a set of hardware parameters into a single-objective minimisation problem. This is fed into a genetic algorithm [41] that minimises the total improvement needed over state-of-the-art performance. These results show there is substantial work left to be done in hardware improvements, and they are summarised in figure 5.

We organise this paper as follows. In section 2, we provide details of the physical setup that we simulate, including physical parameters. Then in section 3, we analyse the two primary results of our numerical analysis discussed briefly above. Section 4 details the trapped-ion model, the client's measurement apparatus, and the genetic algorithm and cost function that define our optimisation procedure. Finally, in section 5 we discuss possible directions for future work.

2. Setup

We simulate a two-party rVBQC setup with a trapped-ion quantum server and a measurement-only client using NetSquid. We investigate the protocol at a metropolitan scale, in which the server and client are

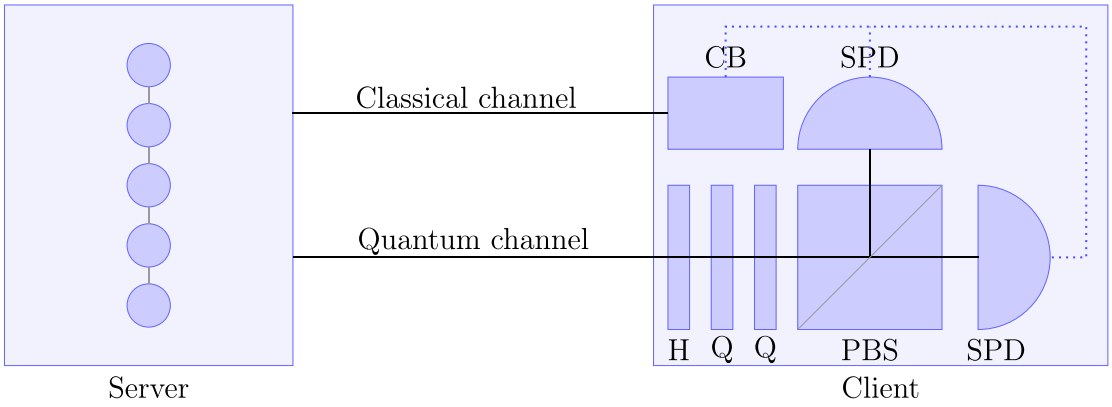


Figure 1. Setup overview of a trapped-ion quantum server (depicted with 5-qubit linear graph state), connected to a client that rotates the polarisation of photons coming in through the quantum channel using half (H) and quarter (Q) waveplates and measures the qubit using a polarising beam splitter (PBS) and single photon detectors (SPDs). The classical channel is used by the client to coordinate with the server. The classical box (CB) is a classical unit that the client uses to choose graph colouring and measurement angles, handle classical communication and perform checks on the results send by the server. The client setup is depicted with more detail to show how it is modelled in the simulation. The exact setup of the server is not shown (i.e. it is kept more general).

separated by 50 km of optical fibre. An overview of the setup is provided in figure 1. The protocol used here, as in [35], assumes a variation wherein the client uses measurements to perform remote state preparation (RSP) on the server. In RSP, a sender measures part of an entangled state and communicates a classical correction to prepare a target state at a receiver.

Below, we will outline how normally rVBQC includes both computation and test rounds, and why our analysis only focuses on test rounds (section 2.1). In sections 2.2 through 2.5 we describe the steps of such a test round, which summarises the protocol of [32] and how this is adapted for our simulations. A visualisation of these steps is given in figure 2. After this, we introduce the parameter sets that are used as input to the simulation (section 2.6) and the metrics on which we base our analysis (section 2.7). This provides necessary context to understand the results as presented in section 3.

2.1. Computation and test rounds

In a normal run of the protocol, one picks a total number of rounds N that are separated into computation rounds and test rounds, as suggested in [32]. The output of the total computation is taken as the majority output of the computation rounds. A formal description of the general protocol can be found in appendix A.

In our analysis, we focus only on test rounds as the computation rounds do not provide additional information about the verifiability of the setup. From the test rounds, we extract the error probability as the fraction of failed rounds. With the resource we consider, a five-qubit linear graph—the test rounds can be accompanied by computation rounds in which any single-qubit gate is performed [42]. Thus, our results are universal for single-qubit gates.

2.2. RSP

The client controls the state of the qubits at the server via RSP. The client can project a qubit located at the server onto a chosen basis by measuring a second qubit, emitted by and entangled with the first qubit (and sent to the client). In this way, the state of the qubits is known to (and determined by) the client, but it is unknown to the server. This is the source of the blindness of this protocol.

In each round, the client randomly chooses the sets of qubits that will become the trap qubits and the dummy qubits, such that all trap qubits are surrounded by dummies. For a five-node graph, one can have either dummy-trap-dummy-trap-dummy or trap-dummy-trap-dummy-trap (other combinations are sub-optimal for trap insertion). The dummy qubits need to be prepared in the standard basis $0/1$ (i.e. on the north or south pole of the Bloch sphere), and the trap qubits need to be prepared in a superposition basis $|\pm_{\theta_v}\rangle = (|0\rangle \pm e^{i\theta_v}|1\rangle)/\sqrt{2}$ (i.e. on the equator of the Bloch sphere) where $\theta_v \in \{k\pi/4\}_{0\leqslant k\leqslant 7}$, with k an integer such that the angle θ_v is randomly and independently chosen for each qubit v.

To start off RSP, the client communicates a description of the graph it wants to prepare to the server. The graph description is the same for all rounds and does not reveal which of the qubits are trap qubits or that a test round is performed at all. This description includes the nodes in the graph I and the edges of the graph G (describing which nodes are connected through CZ gates). Here, we consider a five-qubit linear graph $(I = \{0,1,2,3,4\}; G = \{(0,1),(1,2),(2,3),(3,4)\})$.

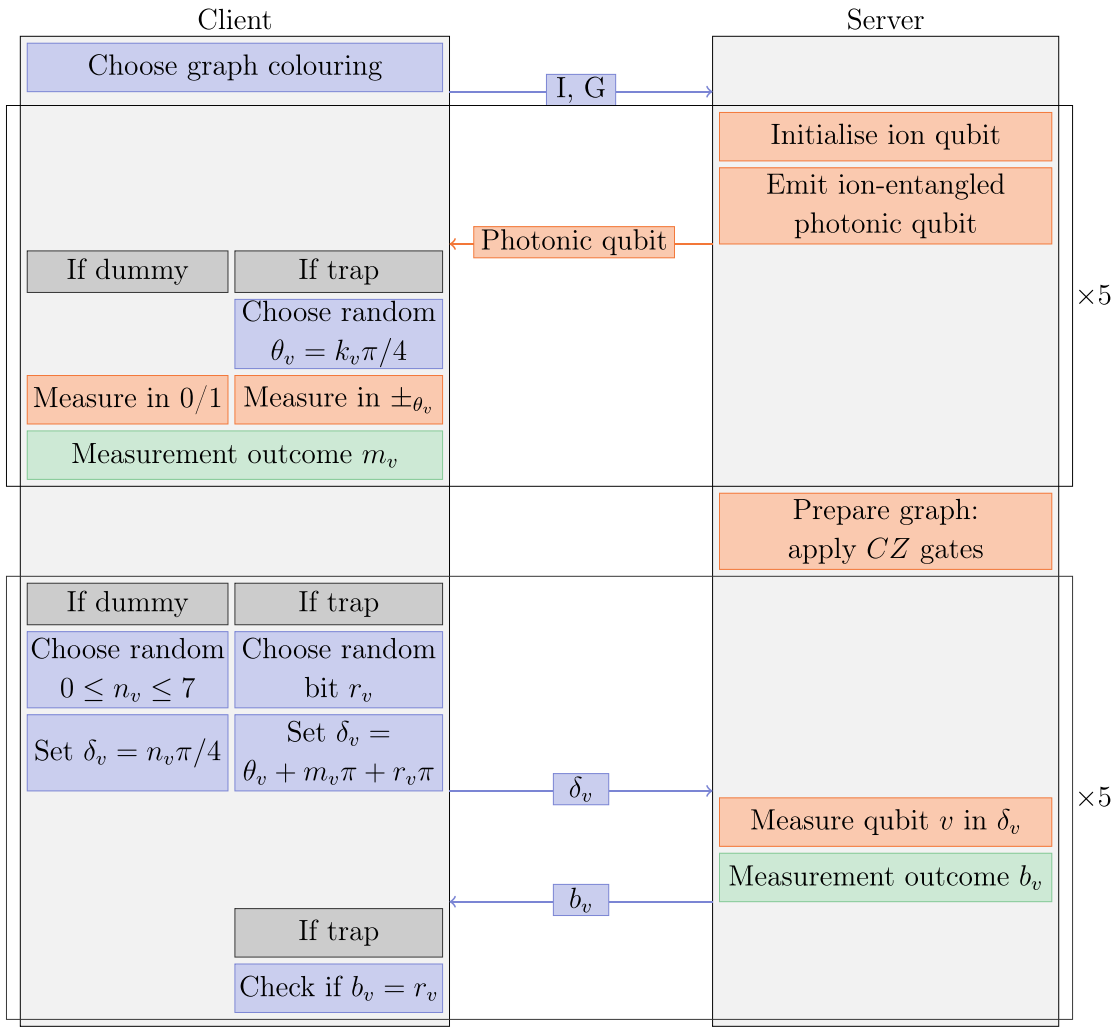


Figure 2. Visualisation of a test round. Depicts tasks (blocks) performed by the client (left) and server (right) along with communication between them (arrows). Blue blocks and arrows represent classical tasks and communication; orange blocks and arrows represent quantum tasks and communication. Grey blocks add comments, and green blocks highlight the measurement outcomes. The test round is depicted in chronological order from top to bottom, where the boxed areas (black outline) are repeated five times, assuming they are successful, as these steps are performed for all qubits in the graph.

After receiving the description, the server emits a polarisation-encoded photonic qubit that is entangled with an ion as $|\psi\rangle = (|0H\rangle + |1V\rangle)/\sqrt{2}$, and this photonic qubit is sent to the client over 50 km of optical fibre. The client then sends back a confirmation of arrival to the server and measures the photon in the standard or superposition basis, depending on whether it is a trap qubit or dummy. The client thereby remotely prepares five qubits on the server that are in the state $|m_v\rangle$ or $|+_{\theta'_v}\rangle = (|0\rangle + e^{i\theta'_v}|1\rangle)/\sqrt{2}$, where $\theta'_v = \theta_v + m_v\pi$, with m_v the outcome of the client's measurement. Note that the server does not know the basis in which the client measures, and therefore it is unaware of the state of the qubits in its memory. Because the server does not know the state of the qubits, it also cannot tell the difference between a computation round (which involves only $|\pm_{\theta_v}\rangle$-qubits) and a test round (which includes dummy qubits) without performing a (malicious) intermediate measurement that has a chance of disturbing the quantum state.

The client performs the superposition basis measurement by rotating the polarisation state of the incoming ion-entangled photonic qubit, then measuring the qubit in the standard basis. Rotating the polarisation state of the photonic qubit can be done by optical elements such as waveplates or electro-optic modulators. A measurement in the standard basis can be performed by a polarising beam splitter followed by single photon detectors (SPDs). For a more detailed description of a possible physical realisation of the client, see [35]. The simulated rotation setup consists of one half waveplate followed by two quarter waveplates, based on [43].

2.3. Cutoff time

In the process of RSP, some qubits will have been prepared and will be sitting in the memory, waiting for the preparation of the remaining qubits. Depending on both server efficiency fibre losses, remotely preparing a

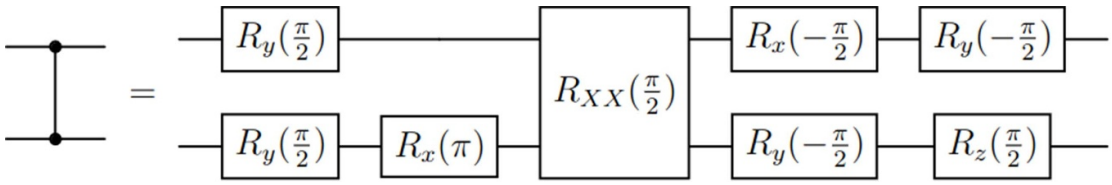


Figure 3. Controlled-Z gate compiled using Qiskit for assumed native gate set of the trapped-ion quantum server. The server is assumed to be able to individually address the ions for rotations (R_x, R_y, R_z), and to be able to perform the bichromatic gate (R_{XX}, also known as Mølmer–Sørensen gate) as the native entangling gate.

qubit might take many tries, meaning that the qubits may have to reside in the memory for a long time. The longer the qubits are in memory, the more they decohere (details in section 4.1). For a lower error probability, we want to limit the time qubits spend in memory, for which we include a cutoff time.

Choosing a good cutoff time is a trade-off between rate and fidelity: a shorter cutoff time gives higher quality qubits as they have suffered from less decoherence, yet more qubits will be discarded, leading to a lower rate. However, since this work only targets an error probability (which depends on the fidelity of the remotely prepared qubits) and does not consider rate, no optimisation can be performed on this aspect. See [44] for examples of optimising this. Instead, we choose a fixed cutoff of half the coherence time.

When any of the qubits in memory have been at the server for longer than the cutoff time, they are discarded and the client is notified. When the server has prepared five qubits, no more qubits will be discarded (i.e. no cutoff is imposed after RSP).

2.4. Graph formation

The client instructs the server to arrange the qubits in the linear graph: edges should be made between qubits $(0,1),(1,2),(2,3)$ and $(3,4)$.

We assume that the ion qubits can be individually addressed for rotations R_x, R_y, R_z. This ability is hardware dependent, it is possible in, for example, [45–47], each of which uses a different mechanism to implement individual addressing. We note that it is sufficient to implement individual addressing for rotations around just one axis together with collective global rotations around an arbitrary axis in the orthogonal plane [48]. We assume the bichromatic gate $R_{XX}(\theta) = \exp(-\mathrm{i}\theta/2\sum_{k<l}\sigma_x^{(k)}\sigma_x^{(l)})$, also known as the Mølmer–Sørensen gate [49, 50], to be the native entangling operation. We use Qiskit [51] to construct a CZ gate using the operations available for the ion trap (figure 3).

In principle, the server does not need to wait for all qubits to be prepared before it starts applying the CZ operations, but this is done in the simulation for simplicity. The time it takes to prepare the graph state (around 150 μs) is short compared to the expected time it takes for the qubits to be remotely prepared (on the order of tens of milliseconds[7]).

2.5. Server measurements and output

Once the graph-state formation is complete, the client instructs the server to measure in $\pm_{\delta_v}$-bases defined by [32]

$$\delta_v = \theta_v + m_v\pi + r_v\pi, \text{for trap qubits,} \tag{1}$$

$$\delta_v \in \{k\pi/4\}_{0\leqslant k\leqslant 7}, \text{for dummy qubits.} \tag{2}$$

Here, θ_v defines the basis used by the client to perform RSP, and m_v is the outcome of the RSP measurement of the photonic qubit entangled with ion qubit v. The random bit r_v is generated by the client to ensure the measurement outcomes appear random. The angles at which the dummy qubits are measured is irrelevant to the client; they just need to be random, so that the server will not be able to identify the qubit as a dummy (which would reveal that the client is executing a test round).

The server communicates the outcome of measuring qubit v to the client as b_v. For all trap qubits, the client checks if $b_v = r_v$. The test round succeeds if this is true for all trap qubits. If any of the trap qubits do not satisfy $b_v = r_v$, the round fails. The round failing or succeeding is the output for each of the test rounds.

[7] The average time it takes to remotely prepare a qubit is calculated as time it takes to perform one attempt divided by the average probability of a photon arriving at the client. The average time is calculated as the distance back and forth (send qubit to client and wait for confirmation) divided by the speed of light in fibre: $2 \times 50\,\mathrm{km} / (c/n) \approx 0.50\,\mathrm{ms}$, with c being the speed of light, and $n \approx 1.5$ being the index of refection of standard telecommunication optical fibre. The arrival probability is calculated as the server efficiency (given in table 2) times the probability of photon transmitting through the fibre: 0.1325 (from table 2) $\times$ ($50\,\mathrm{km} \times 0.2\,\mathrm{dB\,km^{-1}}$) = 0.1. This gives an average probability of photon arriving of 0.01 325. Thus, it takes on average 0.50/0.01 325 = 37.8 ms to remotely prepare a qubit.

Table 1. Parameter set 1: hardware parameters used as simulation input that are not varied over.

Parameter	Value
Channel length	50 km
Photon loss probability in fibre	0.2 dB km^{-1}
Waveplate errors (fast axis tilt/retardation deviation)	0.001[a]
Dark count probability of photon detectors	0.02%[b]
Crosstalk in polarising beam splitter	0.0001
Qubit rotation duration	12 μs [53]
Entangling gate duration	107 μs [53]
Ion initialisation duration	300 ns [54]
Photon emission duration	300 ns [54]
Ion qubit readout duration	100 μs

[a] The waveplate error probability is discussed in section 4.2.
[b] The dark count probability is given as $1 - e^{-R_{\mathrm{dc}}\tau}$ with $R_{\mathrm{dc}} \approx 1500$ Hz the dark count rate for SPDs such as the SPDMA Si Avalanche Photodetector by Thorlabs and $\tau = 12.5$ ns the detection time window such as in the supplementary material of [35].

Table 2. Parameter set 2: baseline of long-distance trapped-ion experiments consistent with the state of the art. We vary over these parameters to find hardware requirements to perform rVBQC. Server efficiency here refers to the total efficiency of preparing an ion-qubit, emitting an ion-entangled photon, coupling to the fibre (combined as the number for 'emit' in the table) and converting its frequency to the telecom C band at 1550 nm ('freq. conversion' in the table). The emission fidelity refers to the fidelity of the ion–photon entangled pair when the photon is emitted.

Parameter	Baseline value
Server efficiency (= emit × freq. conversion)	0.1325 (= 0.53 [57] × 0.25 [58])
Single-qubit gate fidelity	0.99 [59]
Entangling gate fidelity	0.95 [60]
Emission fidelity	0.974 [61]
Coherence time in ms	62 [53]

2.6. Parameter sets

The steps from the previous subsections are simulated in NetSquid to find the error probability (i.e. the fraction of failed test rounds) for a given setup. The simulation takes a set of hardware parameters as input. We divide these hardware parameters into two sets: one set that we do not vary during optimisation (parameter set 1), and one set that we do vary during optimisation (parameter set 2). The first set contains parameters that are inherent to the setup (fibre length), parameters that are established optical components with little room for improvement (fibre loss, crosstalk in beamsplitters, waveplate error probability, detector dark-count rate) and the duration of operations at the ion trap. The last is not optimised over as these timescales are already very small compared to the time required to perform RSP (which is in the order of 20 ms per qubit, two orders of magnitude larger than the timescale of performing gates and readout). This first set is presented in table 1. We do not take into account errors due to polarisation drift in fibre. This effect is assumed to be very small due to corrections with techniques such as the fully automated stabilisation using reference pulses described in [52].

The second set of parameters are the optimisation parameters. These are the parameters we vary in the minimisation methods explained in section 4.3. In table 2 we list these parameters as well as what we consider the 'baseline'. The baseline reflects the current state of the art in long-distance trapped-ion experiments and is used in the minimisation methods to find the difference between what is currently achievable and what is needed to run the protocol at metropolitan distances, as will be explained in section 4.3. We choose this baseline instead of individual values from current record experiments (such as the long coherence time of [55], the high gate fidelities of [45, 47] or the high entanglement fidelity of [56]) because we believe it is more realistic for long-distance experiments. We are actively looking ahead to what we need to achieve rVBQC at a metropolitan scale (i.e. over a 50 km distance), instead of focusing on in-lab experiments, as rVBQC is already feasible in a lab setting [35].

2.7. Error tolerance and requirements

Here, we justify the threshold and explain the two different optimisation procedures we use to come to two different sets of minimised parameters.

In abstract cryptography (i.e. the theory behind the security proof of the rVBQC protocol [32]), security is defined as the indistinguishability between the real-world implementation of the protocol and the ideal, noiseless resource [62]. When the security of a protocol holds over sequential or parallel repetitions with

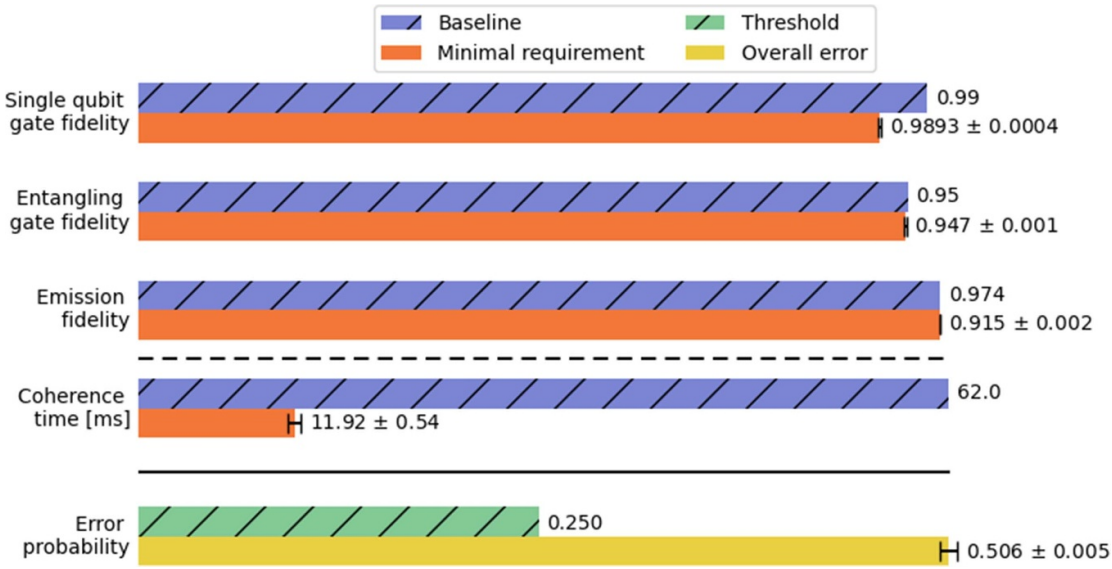


Figure 4. Absolute minimum value a given parameter can have that is consistent with the required maximum error probability of 25%, assuming that the only other imperfection in the system is photon loss in fibre (orange). These minima are compared to the state of the art as given in table 2 (blue/purple, hatched). The dashed black line indicates a change in scaling between the fidelities and the coherence time. Below the solid black line, the overall error with all imperfections in the state of the art (yellow) is compared with the threshold error probability of 25% (green hatched); despite each parameter's baseline value meeting the minimal requirements, together the baseline values are insufficient to run the rVBQC protocol.

other protocols, it is said to be *composably secure*. The rVBQC protocol we study (appendix A) is ϵ-composably-secure (as defined in [32]) with ϵ exponentially small in the number of rounds if the fraction of failed test rounds over total number of test rounds is bounded by 25% [32]. Details for this are provided in section 4.3. The protocol includes classical error correction in the form of a repetition code, but includes no quantum error correction. Note that the 25% error constraint is only present for rVBQC, in non-VBQC the protocol can in principle be performed at any error probability but without any guarantee of the correctness of the outcome.

We can use the error tolerance to find two sets of requirements: absolute minimal hardware requirements and minimal improvements. To find the absolute minimal hardware requirements, we start by setting all hardware parameters to perfect except for photon loss in fibre. That is, for parameter set 1, we set the waveplate errors, dark-count probability and beam splitter crosstalk to 0. The loss in fibre is kept as presented in table 1 at 0.2 dB km^{-1}. For parameter set 2, we set the server efficiency and all fidelities to 1, and we remove the memory decoherence noise model to simulate effectively infinite coherence time. Then one parameter at a time from set 2 is made progressively worse, until the simulated error probability rises above 25%. The last value of the parameter before this happens is taken as the absolute minimal requirement.

For the minimal improvements, we aim to find the least costly improvements needed over the baseline parameters to get the error probability below 25%. The cost of a hardware parameter is given in terms of an improvement factor. This improvement factor quantifies the difference between a given value and the baseline, being 1 for no improvement, and tending to infinity for a parameter tending to perfect (e.g. infinite coherence time, fidelity of 1). More information on the cost function and the improvement factor can be found in section 4.3. The minimal improvements are found by combining the error tolerance with the improvement factors into a single objective optimisation problem and solving it using a genetic algorithm.

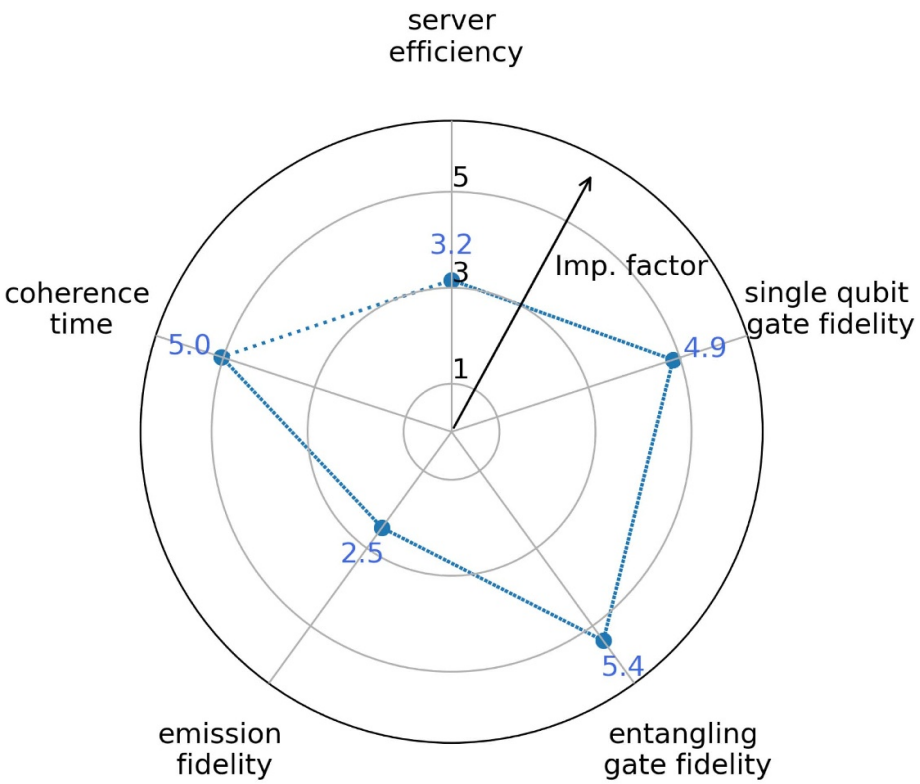
3. Results and discussion

We find requirements for enabling rVBQC with a measurement-only client and a trapped-ion server separated by 50 km of optical fibre, for five-qubit linear graphs.

The absolute minimal requirements found, compared to the baseline of table 2, are shown in figure 4. We see that state-of-the-art ion traps satisfy absolute minimal requirements on a per-parameter basis. However, we also calculate the error probability with the full baseline of parameter set 1. This shows us that all current imperfections combined make it impossible to perform the blind computation securely over 50 km using existing technology, as the corresponding error probability of 51% is twice the requirement of 25%. Note that there is no absolute minimal requirement on the server efficiency, as there is no threshold for rate, only for error probability. If all other parameters are perfect, any server inefficiency combined with fibre attenuation

Parameter	Baseline	Minimally improved value
Server efficiency	0.133	0.393
Single-qubit gate fidelity	0.99	0.997
Entangling gate fidelity	0.95	0.988
Emission fidelity	0.974	0.982
Coherence time [ms]	62	124

(a)



(b)

Figure 5. (a) Minimally improved parameter set required to perform a 5-qubit linear-graph rVBQC protocol on a trapped-ion server and a measurement-only client over 50 km with a 25% error probability threshold. This set minimises the cost function (10), meaning it is the closest to the state-of-the-art baseline of the sets that meet the requirement. (b) Directions along which hardware must be improved to implement a 5-qubit linear graph rVBQC protocol on a trapped-ion server and a measurement-only client over 50 km. The further away the line is from the centre towards a given parameter, the larger the improvement that parameter requires. The improvements are given in terms of an improvement factor k, which tends to infinity as a parameter tends to its perfect value and is 1 for no improvement.

will still lead to a 0% error probability, as the storage of qubits is perfect. (This is not the case for the minimal improvements set, as other parameters will be imperfect. Having imperfect server efficiency then leads to qubits residing in imperfect memory for longer, thereby suffering more decoherence.) It will only lead to lower rate, not considered here. The difference between the minimal requirement and the state of the art appears particularly large for the coherence time. However, we note that the visualisation is skewed as all other parameters are on a scale of zero to one, which is not the case for coherence time. Because of this we chose to separate the coherence time from other parameters with a dashed line, to indicate a different scaling.

We find the minimal improvements, i.e. the set that is closest to the baseline in table 2 satisfying the threshold. The set of hardware parameters that minimises the cost function are given in figure 5(a) and visualised in figure 5(b). The plot in figure 5(b) shows the improvement factor for each parameter in figure 5(a). The further away the line is from the centre towards a given parameter, the larger improvement that parameter requires. From this we can see that comparatively little improvement is needed in terms of

improvement factor for server efficiency and emission fidelity. The server efficiency for the ion trap considered here is enhanced due to the use of a cavity [57].

More improvements are needed for the entangling and single-qubit gate fidelities and for the coherence time. As can be seen from figure 3, many single-qubit gates are executed in order to perform the *CZ* gate used for creating the graph state, which makes any imperfections in the execution of such single-qubit gates more impactful. We do note that the number of single-qubit gates may be reduced by optimising the graph state creation and perhaps absorbing some of the gates into the measurement bases, which would reduce the overall errors and hence lessen the need for improvement on this parameter. This is not the case for the entangling-gate fidelity.

Notably, a lower improvement factor does not always means a value is easier to obtain, this lies in the problem of finding a meaningful cost function. In this case, most values have been obtained in separate optimised experiments, but this is not true for the server efficiency. The baseline used for this optimisation is already optimised for server efficiency as it comes from communication experiments, such that further improvement might be more challenging. Therefore, while the improvement factor might help in visualisation, the true values given in table of figure 5(a) might give a clearer idea of what improvements are needed.

Genetic algorithms do not guarantee to find a global minimum, instead several local minima were found, of which the one presented here was the lowest. Other solutions are roughly equivalent but might give slightly more importance on improving one parameter over the other. An alternative solution is given in appendix B for comparison, and other datasets can be found in [63].

4. Methods

Here, we discuss some details of how the trapped-ion server and client apparatus are modelled in our NetSquid simulation. We also outline the minimisation method used to determine the requirements identified in the previous section.

Though in principle the optimisation can be executed using a different simulator, the choice for using NetSquid as opposed to other quantum network simulators is threefold. First, it is well suited for this type of modelling, as it is a dedicated quantum network simulator that simulates the hardware layer on an appropriate scale. Second, an ecosystem of open-source user-contributed libraries has developed around NetSquid, providing us with useful tools and examples [64, 65]. We have been able to contribute back to this ecosystem by integrating our own library NetSquid-TrappedIons [38]. Lastly, some of the authors of this work having prior experience with NetSquid made it a natural choice.

4.1. Trapped-ion modelling

We model the trapped-ion server using the NetSquid-based library NetSquid-TrappedIons [38]. This library was first used in [40] and is here introduced in more detail.

We model the decoherence of trapped-ion qubits over time using a collective Gaussian dephasing process [66], which can be rewritten as [40]

$$\rho \to \int_{-\infty}^{\infty} K_r \rho K_r^\dagger p(r)\,\mathrm{d}r, \tag{3}$$

with

$$K_r = \exp\left(-\mathrm{i}r\frac{t}{\tau}\sum_{j=1}^{n}\sigma_z^{(j)}\right) \tag{4}$$

and

$$p(r) = \frac{1}{\sqrt{2\pi}}e^{-r^2/2}. \tag{5}$$

Here, $\sigma_z^{(j)}$ is the Pauli Z operator acting on qubit j, τ is the coherence time of the ion qubit and t is the amount of time that has passed. In addition, K_r is the unitary part of the Kraus operator $K_r\sqrt{p(r)\mathrm{d}r}$ satisfying Kraus' theorem: $\int_{-\infty}^{\infty} K_r^\dagger K_r p(r)\mathrm{d}r = \int_{-\infty}^{\infty} p(r)\mathrm{d}r = 1$. By writing the model this way, one makes the following interpretation explicit: all the qubits undergo Z rotations at a constant rate of $2r$ per time interval τ, where r is a random variable with probability distribution $p(r)$. We note that the model is 'collective' in the sense that there is correlated noise between all the qubits in the same ion trap (they all undergo the same random rotation), and 'Gaussian' in the sense that the probability that no dephasing error took place

decreases with a Gaussian profile over time, which is a consequence of r being normally distributed. The noise process is non-Markovian, which poses a challenge when modelling it in a discrete-event simulator like NetSquid. In NetSquid-TrappedIons, this problem is solved by sampling a value for r from $p(r)$ each time the ion qubit is reinitialised, then evolving the qubits over time using the corresponding unitary operator K_r.

The emission of entangled photons from an ion qubit is modelled in NetSquid-TrappedIons as the creation of a photon that has a polarisation degree of freedom that is maximally entangled with the state of the ion used in the emission, $|\psi\rangle = (|0H\rangle + |1V\rangle)/\sqrt{2}$, followed by the application of a single-qubit depolarising channel on the photon's polarisation. This results in a Werner state of the form $\frac{3}{4}F|\psi\rangle\langle\psi| + \frac{4F-1}{3}\frac{\mathbb{1}}{4}$, where F is the fidelity of the state with respect to the perfect state $|\psi\rangle$.

4.2. Client apparatus modelling

In simulating the client depicted in figure 1, the effect of the waveplates is given by the multiplication of the waveplate Jones matrices with the state vector [67]. The relative phase retardation induced between the fast axis and the slow axis is $\delta = \pi/2$ for a quarter waveplate and $\delta = \pi$ for a half waveplate. The fast axes of the waveplates also have an angle of ξ radians with respect to the x-axis (which is along the plane of polarisation for linearly polarised light), determining the specific rotation that is implemented. However, errors in the setup can influence the retardation, giving a retardation deviation of $\Delta\delta$. It is also possible to have a deviation in the angle ξ leading to $\Delta\xi$. With this, we can write the Jones matrices in general form [68] to include the errors as

$$U(\delta',\xi') = e^{-i\delta'/2}\begin{pmatrix} \cos^2(\xi') + e^{i\delta'}\sin^2(\xi') & (1-e^{i\delta'})\cos(\xi')\sin(\xi')) \\ (1-e^{i\delta'})\cos(\xi')\sin(\xi')) & \sin^2(\xi') + e^{i\delta'}\cos^2(\xi') \end{pmatrix}, \tag{6}$$

where $\delta' = \delta + \Delta\delta$ (with $\delta = \pi/2$ for a QWP, and $\delta = \pi$ for a HWP) and $\xi' = \xi + \Delta\xi$. Together, $\Delta\delta$ and $\Delta\xi$ lead to estimated waveplate error probability as given in table 1. The waveplates are implemented as a custom operation in NetSquid according to the Jones matrix.

Simon and Mukunda [43] gives an overview of general qubit rotations in terms of these fast axis settings. We can set the fast axes to correspond to a measurement in the $|\pm_\theta\rangle$-basis as

$$\begin{aligned} \xi_1 &= 0; \\ \xi_2 &= \theta/2; \\ \xi_3 &= \theta/4 - 3\pi/4. \end{aligned}$$

4.3. Minimisation methods

In our analysis, we use one target metric: the error tolerance of 25%, which is a bound on the fraction of test rounds that are allowed to fail while still being ϵ-composably secure. The exact value of ϵ is not considered in this analysis, apart from that it can be made exponentially small by increasing the number of test rounds. In [32] the fraction of failed test rounds w over the total number of test rounds t is bounded by

$$w/t < \frac{1}{k}\frac{2p-1}{2p-2}. \tag{7}$$

Here, k is the principal colouring of the computation graph (which is the smallest number of 'colours' or labels one can give to the nodes in a graph such that no two neighbouring nodes have the same colour; see, for example, [69]) and p is the inherent error probability of the bounded-error quantum computation. Assuming a computation for which $p = 0$ we require the error probability to be below $1/2k$. The one-dimensional graph for single-qubit rotations used in this paper is two-colourable, which gives a maximum error tolerance of 25%.

We look for absolute minimal hardware requirements by setting all but one parameter to perfect aside from fibre attenuation. For the coherence time, perfect means removing the collective dephasing noise model (section 4.1) from the ions in the trap. We then sweep over the imperfect parameter to find where the error probability due to this imperfection crosses the threshold.

To find the crossing point, we do an initial global search with a small number of test rounds per point (to limit computation time) to find the approximate regime in which the error probability would pass the threshold. Once the region is located, the search is focused by taking the closest point above and below the threshold, halving the distance to their mean and running the simulation again for these points with a larger number of rounds. This process is repeated until the error-probability confidence interval of the points crosses the threshold of 25%. (This is similar to the bisection method used in root finding). The confidence intervals are determined by Hoeffding's bound [70] as $\sqrt{\ln(2/0.05)/2t}$, with t the number of test rounds. The focused search is executed with 70 000 points in order to have a confidence level of 95% in an interval $\pm$

Table 3. Probabilities of no-imperfection: re-scaling parameters from zero to one. This is used to associate an improvement factor from the baseline to each parameter that is dimensionless and thus comparable.

Parameter	p_{NI}
Server efficiency η	η
Coherence time T_c	e^{-t^2/T_c^2}
(Gate and entangled state) fidelity F	$\frac{1}{3}(4F-1)$

0.005 for the error probability. The minimal requirements given in figure 4 are then extracted from the closest points $((x_1,y_1),(x_2,y_2))$ by a linear interpolation at the threshold ($y=0.25$) as $x=(y-y_1)*(x_2-x_1)/(y_2-y_1)$. The error in these estimates is found by applying the same interpolation to the edges of the error probability confidence interval.

Next, we find the set of minimal improvements. That is, from a given baseline (table 2), what parameters allows us to fulfill the constraint on the error probability with the least improvement? In order to quantify the cost of improving a parameter by a certain amount, we define hardware costs H_c and a cost function C, which combines the hardware costs with our constraint on the error probability to give a single-objective minimisation problem as done in [39, 40, 71, 72]. We then employ a genetic algorithm using a workflow manager called YOTSE [73] to find the set of hardware parameters that minimises the cost function. This optimisation was run on SURF's[8] high-performance-computing cluster Snellius (Platinum 8360Y CPU @ 2.4 GHz, maximum of 480 GB RAM) and on a workstation featuring an Intel Xeon Gold 6230 CPU @ 2.10 GHz and 188 GB of DDR4 RAM, taking around 8000 core-hours per optimisation run of 20 generations.

To have a consistent way of calculating hardware cost, we associate a probability of no-imperfection $p_{NI}(b_i)$ to each of the N baseline hardware parameters $b_i \in B$, where B is the baseline set of N hardware parameters. This scales all parameters from 0 to 1, where 1 means a perfect setting (e.g. infinite coherence time, 100% efficiency). We can improve upon this baseline with an improvement factor k to find $p_{NI}(x_i)=\sqrt[k]{p_{NI}(b_i)}$. These p_{NI} are then summed over for all hardware parameters x_i to find the total hardware cost $H_c(X)$ of a setup with hardware set $X=\{x_i\}_{0<i\leqslant N}$ with respect to the baseline B as

$$H_c(X)=\sum_{i=1}^{N}\frac{\ln\{p_{NI}(b_i)\}}{\ln\{p_{NI}(x_i)\}}. \tag{8}$$

This is equivalent to summing over the improvement factor of each parameter. The probabilities of no-imperfection are defined for the optimisation parameters as in table 3.

For derivations and further explanation of the probabilities of no-imperfection, see supplementary note 6 of [40]. Note that the variable t in the probability of no-imperfection of the coherence time, indicating the timescale over which qubits decohere, does not influence the hardware cost, as

$$\frac{\ln(p_{NI}(b_i))}{\ln(p_{NI}(x_i))}=\frac{\ln\left(e^{-t^2/T_c^2}\right)}{\ln\left(\sqrt[k]{e^{-t^2/T_c^2}}\right)}=\frac{-t^2/T_c^2}{-t^2/kT_c^2}=k. \tag{9}$$

We now combine our requirement of having an error probability below 25% with the hardware cost to find the total cost of a set of parameters. We want the cost assigned to a set of parameters to be very high when the constraint is not met, and to be lower the closer parameter sets are to the baseline assuming that the constraint is met. A function to capture this behaviour, similar to what is used in [39, 40, 71, 72], is

$$C=w_1\left(1+(w/t-1/(2k))^2\right)\Theta(w/t-1/(2k))+w_2Hc(x_1,\ldots,x_N), \tag{10}$$

where $\Theta(x)$ is the Heaviside step function and w_1 and w_2 are the weights of the objectives. We choose $w_1 \gg w_2$ in order for the function to reflect that it is much more important to satisfy the error probability requirement than to minimise hardware cost, i.e. we do not care about the hardware cost as long as the requirement is not met.

The genetic algorithm is implemented as follows: for all parameters, a number of points are drawn from a range between the baseline (table 2) and their perfect value (except for the coherence time, which is capped at 1 s), i.e. for which $p_{NI}=1$. We initially draw 3 points for server efficiency, 4 for coherence time, 2 for single-qubit fidelity, 3 for entangling gate fidelity and 2 for emission fidelity, meaning the initial population

[8] SURF is a collaborative organisation for IT in Dutch education and research.

is formed by $3 \times 4 \times 2 \times 3 \times 2 = 144$ sets of parameters. The number of points which are drawn for each parameter is based on the size of the range baseline to perfect for that parameter (e.g. there are more possible values that the entangling gate fidelity can take on than the single-qubit gate fidelity can, as it is further from perfect; we therefore initially draw more points at random from the entangling gate fidelity distribution). From this initial population, the lowest-costing eight 'parents' are taken to recombine with a mutation probability of 0.2 into a new generation. This is repeated over twenty generations. Each point consists of 20 000 test rounds for a confidence interval around the error probability of ± 0.0096. The set of parameters with minimal cost is then fed into a local search algorithm, who decreases the cost of each parameter slightly until the error probability requirement is no longer met. The local search is done with 70 000 rounds for a confidence interval of ± 0.0051. The outcome of this is a set of parameters that is minimal in the sense that further parameter adjustments that lower the cost of any parameter at that point will result in the requirement not being met.

Due to the limits of the search space of the parameters, the lowest value for the hardware cost per parameter is one, when the value of this parameter is equal to that of the baseline. Therefore H_c is bounded from below by 5 (as 5 parameters are considered). The cost does not have an upper bound, and tends to infinity for any parameter tending to its perfect value. In practice, however, the cost does not extend much past w_1.

5. Future work

One could modify the cost function (10) to include a constraint on the rate at which the computation or test rounds can be performed. This might change the directions along which the hardware should be improved (i.e. it changes the set of minimal improvements) as parameters such as server efficiency become more important, and conversely the coherence time would become less important. This could be done in the same workflow, by changing the cost function and choosing a rate constraint. We have however chosen not to include this, as there is no immediate clear goal in rate, whereas there is a clear goal in error probability (25%).

The constraint on the error probability of 25% we consider in this paper is a theoretical limit. In reality, an error probability of 25% would require an impractically large number of rounds in order to find a desirable ϵ for security. Instead of setting a constraint on a minimum rate at which the computations can be performed, we might want to find a different metric to target than just the rate and error probability. Having a lower error probability would allow one to perform fewer rounds of the protocol, thereby finishing the total computation faster. An option could be to consider the rate of successfully completed computations, which depends on the error probability, as this determines the number of rounds to be performed, as well as on the rate at which these rounds are performed. This is beyond the scope of this paper.

One could also consider requirements for larger graph states. The current analysis considers a universal resource for single-qubit rotations, but a universal resource for any quantum computation (i.e. including two-qubit gates), such as a brickwork state [16], would require more qubits. This will lead to more stringent requirements on the hardware. In principle, the same framework used in this paper could be extended to larger graphs such as the brickwork state, but the current state of the code would make the computation time quite a bit longer. We estimate that a 10-qubit graph would take about 2.5 times as long as the 5-qubit graph, the method for this estimation is described in appendix C. This means that the full optimisation procedure (i.e. 20 generations of the genetic algorithm) is estimated to take about 20 000 core-hours to complete for a 10-qubit version. The code could be sped up by including a framework similar to NetSquid's entanglement generation Magic [74], which offers simulation speedup through state insertion. An equivalent to this for RSP is currently in development.

The protocol could be optimised by limiting the number of single-qubit gates used in the formation of the graph state, either by considering a full-graph optimisation or by absorbing some of the single-qubit gates into the measurement bases. We also note that, depending on the graph state, not all qubits need to be 'alive' in the memory at the same time. The first qubit can be measured before the last qubit is initiated, as long as they are not nearest-neighbours in the graph. In addition, if additional memory qubits are available, the RSP phase could be parallelised by sending ion-entangled photonic qubits from different memory positions successively without waiting for a heralding signal in between.

Data availability statement

The code used to generate, process and plot the data is available at [75].

The data that support the findings of this study are openly available at the following URL/DOI: https://doi.org/10.4121/C98E631C-A6D3-411E-AD4C-4DECF6943F57.V1.

Acknowledgments

We thank Harold Ollivier and Maxime Garnier for useful discussions on rVBQC. We thank Viktor Krutianskii for useful discussions on trapped-ion devices and for sharing experimental data and parameters. We thank David Maier for providing technical support for the code and execution thereof. We thank Kian van der Enden for critical reading of the manuscript. This project (QIA-Phase 1) has received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement No. 101102140.

Author contributions

J v D led the development and execution of the simulation of the protocol, gathered and analysed the results, and prepared this manuscript. G A contributed to development of hardware models of trapped ions and prepared section 4.1 of this manuscript. Tz B P and J A S provided feedback and discussions throughout the project. F F d S provided technical support and guidance. T E N helped in the technical understanding of the ion traps and the parameters. All authors revised the manuscript. S W conceived and supervised the project.

Appendix A. Formal protocol

Based on [32], as in [35].

Clients inputs: Angles $\{\phi_v\}_{v\in V}$ for all qubits (*vertices*) V, determining the gate(s); a graph G; a flow f on G, determining the order of measurements.

Protocol:

1. The client chooses uniformly at random a partition (C, T) of the set of indices of all the rounds in the protocol $N, C\cap T=\emptyset$, with C and T the set of indices for computation and test rounds, respectively.
2. For all rounds $n\in N$ the client and server perform the following subprotocol (the client may send a message *redo n* to the server before step 2.3, or the server may send it at any time. Both parties then restart round n with fresh randomness, for more information about this redo feature, see [32]):
   - 2.1. If $n\in T$ (test round), the client chooses uniformly at random a colour K_n to define the set of trap vertices for this test round. (The colouring of a graph refers to a way of labelling the nodes such that no neighboring nodes have the same colour, this ensures that no two traps are connected through an edge).
   - 2.2. For all $v\in V$ (i.e. for all qubits): # RSP
     - 2.2.1. The server prepares a bell pair $|\psi\rangle=(|00\rangle+|11\rangle)/\sqrt{2}$ and sends half of it to the client along with a classical ID.
     - 2.2.2 a. When the ID arrives at the client, and the qubit also arrived, the client sends a classical confirmation back to server and performs a measurement yielding outcome m_v. The measurement basis is chosen as:
       - (i) If $n\in T$ and $v\notin K_n$, measure in the standard basis (i.e. prepare a dummy qubit).
       - (ii) If $n\in C$ (computation round) or if $n\in T\wedge v\in K_n$ (trap qubit in test round), measure in $\pm_{\theta_v}$-basis, with $\theta_v\in\{k\pi/4\}_{0\leqslant k\leqslant 7}$ randomly chosen, (i.e. prepare $|\pm_{\theta_v}\rangle$).
     - 2.2.2 b. When ID arrives at client, if qubit did not arrive: client sends classical 'lost' message to server. The server removes the qubit from its memory and goes back to step 2.2.1.
   - 2.3. The client sends description of G, the server performs a CZ gate between all qubits that share an edge according to G (i.e. the server constructs the graph state).
   - 2.4. For all $v\in V$ the client and server perform the following subprotocol:
     - 2.4.1. The client instructs the server to measure in a $\pm_{\delta_v}$ basis defined by:
       - (i) If $n\in C$:

$$\delta_v=\phi'_v+\theta_v+m_v\pi+r_v\pi, \tag{A.1}$$

where $r_v \in_R \{0,1\}$ is chosen uniformly at random. The angle ϕ'_v is defined as

$$\phi'_v = (-1)^{s_{X,v}}\phi_v + s_{Z,v}\pi, \tag{A.2}$$

with

$$s_{X,v} = \bigoplus_{l\in S_{X,v}} s_l,\ s_{Z,v} = \bigoplus_{l\in S_{Z,v}} s_l, \tag{A.3}$$

where $\bigoplus_{l\in S_{X(Z),v}}$ represents a modulo 2 summation over the X (Z) dependency set for qubit v. The dependency sets are defined by $S_{X,v} = f(v-1)$ and $S_{Z,v} = \{l : v \in N_G(f(l))\}$, with N_G referring to the neighbors of a qubit, which are all other qubits connected to it through an edge. These adaptations to the measurement angles eliminate the need for bit flip and phase corrections in between the measurements. For more background on how the measurement angles are determined (ϕ_v) and adapted ($s_{X,v}, s_{Z,v}$), see [36].

(ii) If $n \in T \wedge v \in K_n$ (trap):

$$\delta_v = \theta_v + m_v\pi + r_v\pi, \tag{A.4}$$

i.e. the qubit is being measured in the basis in which it is prepared.

(iii) If $n \in T \wedge v \notin K_n$ (dummy):

$$\delta_v \in \{k\pi/4\}_{0\leqslant k\leqslant 7}, \tag{A.5}$$

is randomly chosen.

2.4.2. The server measures in the basis defined by the client and and sends back the measurement outcome b_v.

2.5. For all $\{v : n \in T \wedge v \in K_n\}$ the client verifies that $b_v = r_v \oplus d_v$, where $d_v = \bigoplus_{i\in N_G(v)} d_i$ is the sum over the measurement outcomes of the neighbouring dummies of qubit v. If this is false for any trap qubit in the test round, the test round fails. If the number of test rounds exceeds a certain fraction w/t, the client aborts. Here, $w/t < \frac{1}{k}\frac{2p-1}{2p-2}$, introduced in (7), is the error threshold to guarantee variability and correctness.

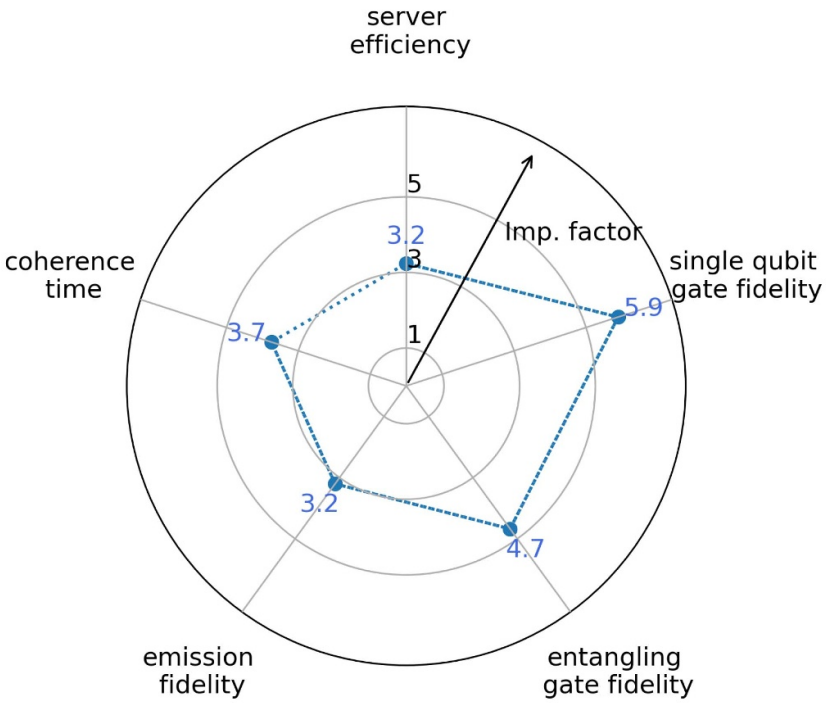
2.6. For all $n \in C$, let y_c be the classical output of computation round c, the clients checks for a majority output, i.d. checks if there exists a y such that $|\{y_c : y_c = y\}| > |C|/2$. If there is a majority output, this y is taken as the protocol output and the client sends an OK to the server.

Appendix B. Alternative route

Multiple sets of parameters can reach the error probability threshold of 25% for a similar cost. In section 3 we give a cost-minimised set of parameters (figure 5) and discuss what what variations in parameters would yield a similar result. The minimised results show less improvement needed in server efficiency and emission fidelity and more in coherence time, entangling and single qubit gate fidelity. How much improvement is needed in these three main objectives varies slightly over different optimisation outcomes. In figure B1, we provide an alternative optimisation outcome compared to the one in the main text to support this observation. In particular, this minimisation puts slightly more emphasis on the single qubit gate fidelity and a little less on the coherence time and entangling gate fidelity compared to the set given in the main text. In addition, the emission fidelity requires more improvement compared to the solution presented in the main text.

Parameter	Baseline	Minimally improved value
Server efficiency	0.133	0.594
Coherence time [ms]	62	103
Single qubit gate fidelity	0.99	0.998
Entangling gate fidelity	0.95	0.986
Ion-photon entanglement fidelity	0.974	0.988

(a)



(b)

Figure B1. (a) Alternative set of parameter values required to perform a 5-qubit linear graph rVBQC protocol on a trapped-ion server and a measurement-only client over 50 km. The requirement for protocol to succeed is having an error probability below 25%. This set minimises the cost function (10), meaning it is closest to the state-of-the-art baseline. A visual representation of these parameters in terms of an improvement factor is given in the bottom figure. (b) Directions along which hardware must be improved to perform a 5-qubit linear graph rVBQC protocol on a trapped-ion server and a measurement-only client over 50 km. The further away the line is from the centre towards a given parameter, the larger improvement that parameter requires. The improvements are given in terms of an improvement factor k, which tends to infinity as a parameter tends to its perfect value and is 1 for no improvement. More information on this can be found in section 4.3.

Appendix C. Estimation of runtime for larger graph states

The simulation currently simulates every attempt on RSP; also the failed ones. When the size of the graph gets larger (i.e. it contains more qubits) we will both need to have success more often and, in particular, they all need to happen within the cutoff time, which is half of the coherence time. This will take longer to simulate. The full optimisation procedure (i.e. 20 generations of the genetic algorithm) has not been run for larger graph sizes. To make an estimate of how long this would take, we can compare how long the 5-qubit optimisation procedure took to the fractional runtime of larger graphs at the baseline. When computation B takes twice as long to perform as computation A, the fractional runtime of B compared to A is 2. The optimisation for the 5-qubit graph took roughly 8000 core-hours, a computation with a fractional runtime of 2 compared to the 5-qubit graph would then take roughly 16 000 core-hours.

To make an estimate of the fractional runtime, we ran the baseline parameters for multiple graph sizes for 5000 test rounds, recorded the time it took to finish and compared this to the time it took to finish the same calculation with a 5-qubit graph. The result of this can be found in figure C1. The fractional runtime seems

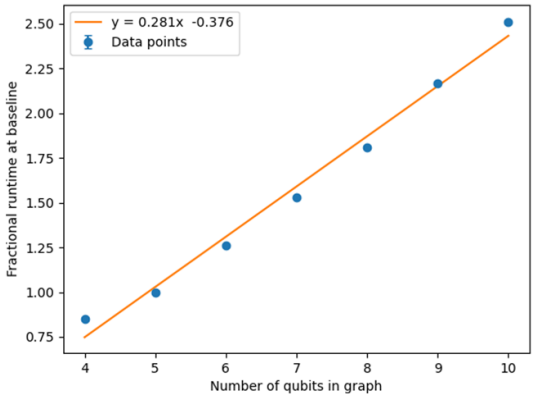


Figure C1. Estimation of the fractional runtime (y-axis) for calculations including a certain number of qubits in the graph (x-axis) compared to the calculation involving a 5-qubit graph. Plot includes standard error of the mean for 5000 points per graph size, error bars smaller than markers (blue). Also plotted with linear fit (orange). This can be used to estimate how long the full optimisation procedure will take.

to increase roughly linearly with increasing number of qubits in the graph, so a linear fit was included to possibly extend this logic to slightly larger graphs. The linear fit results in the relation $y = 0.281x - 0.376$ with x the number of qubits in the graph and y the fractional runtime. However, we do not expect this relation to be linear for all graph sizes as we expect that the probability of all qubits being remotely prepared withing the cutoff time will go to zero more quickly for larger graphs. Additionally, this is a rough estimate as it is only fully applicable to the hardware parameters at baseline, whereas the optimisation procedure will have different parameter sets, though we hope it provides a useful estimate nonetheless.

ORCID iDs

J van Dam https://orcid.org/0009-0004-6593-6974
F Ferreira da Silva https://orcid.org/0000-0003-3642-4350
T E Northup https://orcid.org/0000-0002-1071-2218

References

[1] Feynman R P 1982 Simulating physics with computers *Int. J. Theor. Phys.* **21** 467–88
[2] Shor P W 1994 Algorithms for quantum computation: discrete logarithms and factoring *Proc. 35th Annual Symp. on Foundations of Computer Science* (IEEE) pp 124–34
[3] Grover L K 1997 Quantum mechanics helps in searching for a needle in a haystack *Phys. Rev. Lett.* **79** 325
[4] Arute F *et al* 2019 Quantum supremacy using a programmable superconducting processor *Nature* **574** 505–10
[5] Ebadi S *et al* 2021 Quantum phases of matter on a 256-atom programmable quantum simulator *Nature* **595** 227–32
[6] Madsen L S *et al* 2022 Quantum computational advantage with a programmable photonic processor *Nature* **606** 75–81
[7] Kim Y *et al* 2023 Evidence for the utility of quantum computing before fault tolerance *Nature* **618** 500–5
[8] Moses S A *et al* 2023 A race-track trapped-ion quantum processor *Phys. Rev.* X **13** 041052
[9] Global Future Council on Quantum Computing 2022 State of quantum computing: building a quantum economy (available at: www3.weforum.org/docs/WEF_State_of_Quantum_Computing_2022.pdf)
[10] Martin M J, Hughes C, Moreno G, Jones E B, Sickinger D, Narumanchi S and Grout R 2022 Energy use in quantum data centers: scaling the impact of computer architecture, qubit performance, size and thermal parameters *IEEE Trans. Sustain. Comput.* **7** 864–74
[11] Parker E and Vermeer M J D 2023 Estimating the energy requirements to operate a cryptanalytically relevant quantum computer (arXiv:2304.14344)
[12] Soeparno H and Perbangsa A S 2021 Cloud quantum computing concept and development: a systematic literature review *Proc. Comput. Sci.* **179** 944–54
[13] Sheng Y-B and Zhou L 2017 Distributed secure quantum machine learning *Sci. Bull.* **62** 1025–9
[14] Zhou X and Qiu D 2021 Blind quantum machine learning based on quantum circuit model *Quantum Inf. Process.* **20** 363
[15] Li W, Lu S and Deng D-L 2021 Quantum federated learning through blind quantum computing *Sci. China Phys. Mech. Astron.* **64** 100312
[16] Broadbent A, Fitzsimons J and Kashefi E 2009 Universal blind quantum computation *2009 50th Annual IEEE Symp. on Foundations of Computer Science* (IEEE) pp 517–26
[17] Morimae T and Fujii K 2013 Blind quantum computation protocol in which alice only makes measurements *Phys. Rev.* A **87** 050301
[18] Fitzsimons J F 2017 Private quantum computation: an introduction to blind quantum computing and related protocols *npj Quantum Inf.* **3** 23

[19] Li Q, Liu C, Peng Y, Yu F and Zhang C 2021 Blind quantum computation where a user only performs single-qubit gates *Opt. Laser Technol.* **142** 107190
[20] Morimae T and Fujii K 2013 Secure entanglement distillation for double-server blind quantum computation *Phys. Rev. Lett.* **111** 020502
[21] Sheng Y-B and Zhou L 2015 Deterministic entanglement distillation for secure double-server blind quantum computation *Sci. Rep.* **5** 7815
[22] Li Q, Chan W H, Wu C and Wen Z 2014 Triple-server blind quantum computation using entanglement swapping *Phys. Rev.* A **89** 040302
[23] Quan J, Li Q and Li L 2023 Verifiable blind quantum computation with identity authentication for multi-type clients *IEEE Trans. Inf. Forensics Secur.* **19** 1687–98
[24] Raussendorf R and Briegel H J 2000 Quantum computing via measurements only (arXiv:quant-ph/0010033)
[25] Briegel H J, Browne D E, Dür W, Raussendorf R and Van den Nest M 2009 Measurement-based quantum computation *Nat. Phys.* **5** 19–26
[26] Gheorghiu A, Kapourniotis T and Kashefi E 2019 Verification of quantum computation: an overview of existing approaches *Theory Comput. Syst.* **63** 715–808
[27] Fitzsimons J F and Kashefi E 2017 Unconditionally verifiable blind quantum computation *Phys. Rev.* A **96** 012303
[28] Hayashi M and Morimae T 2015 Verifiable measurement-only blind quantum computing with stabilizer testing *Phys. Rev. Lett.* **115** 220502
[29] Mahadev U 2018 Classical verification of quantum computations *2018 IEEE 59th Annual Symp. on Foundations of Computer Science (FOCS)* (IEEE) pp 259–67
[30] Preskill J 2018 Quantum computing in the NISQ era and beyond *Quantum* **2** 79
[31] Sheng Y-B and Zhou L 2018 Blind quantum computation with a noise channel *Phys. Rev.* A **98** 052343
[32] Leichtle D, Music L, Kashefi E and Ollivier H 2021 Verifying BQP computations on noisy devices with minimal overhead *PRX Quantum* **2** 040302
[33] Barz S, Kashefi E, Broadbent A, Fitzsimons J F, Zeilinger A and Walther P 2012 Demonstration of blind quantum computing *Science* **335** 303–8
[34] Greganti C, Roehsner M-C, Barz S, Morimae T and Walther P 2016 Demonstration of measurement-only blind quantum computing *New J. Phys.* **18** 013020
[35] Drmota P *et al* 2024 Verifiable blind quantum computing with trapped ions and single photons *Phys. Rev. Lett.* **132** 150604
[36] Danos V, Kashefi E and Panangaden P 2007 The measurement calculus *J. ACM* **54** 8–es
[37] Coopmans T *et al* 2021 Netsquid, a network simulator for quantum information using discrete events *Commun. Phys.* **4** 164
[38] Avis G 2023 Netsquid trapped-ions snippet (available at: https://docs.netsquid.org/snippets/netsquid-trappedions/)
[39] da Silva F F, Torres-Knoop A, Coopmans T, Maier D and Wehner S 2021 Optimizing entanglement generation and distribution using genetic algorithms *Quantum Sci. Technol.* **6** 035007
[40] Avis G, da Silva F F, Coopmans T, Dahlberg A, Jirovská H, Maier D, Rabbie J, Torres-Knoop A and Wehner S 2023 Requirements for a processing-node quantum repeater on a real-world fiber grid *npj Quantum Inf.* **9** 100
[41] Mitchell M 1996 *An Introduction to Genetic Algorithms* (MIT Press)
[42] Danos V, Kashefi E and Panangaden P 2005 Parsimonious and robust realizations of unitary maps in the one-way model *Phys. Rev.* A **72** 064301
[43] Simon R and Mukunda N 1990 Minimal three-component SU(2) gadget for polarization optics *Phys. Lett.* A **143** 165–9
[44] Rozpędek F, Goodenough K, Ribeiro J, Kalb N, Caprara Vivoli V, Reiserer A, Hanson R, Wehner S and Elkouss D 2018 Parameter regimes for a single sequential quantum repeater *Quantum Sci. Technol.* **3** 034002
[45] Gaebler J P *et al* 2016 High-fidelity universal gate set for $^{9}Be^{+}$ ion qubits *Phys. Rev. Lett.* **117** 060505
[46] Pogorelov I *et al* 2021 Compact ion-trap quantum computing demonstrator *PRX Quantum* **2** 020343
[47] Ballance C J, Harty T P, Linke N M, Sepiol M A and Lucas D M 2016 High-fidelity quantum logic gates using trapped-ion hyperfine qubits *Phys. Rev. Lett.* **117** 060504
[48] Schindler P *et al* 2013 A quantum information processor with trapped ions *New J. Phys.* **15** 123012
[49] Sørensen A and Mølmer K 1999 Quantum computation with ions in thermal motion *Phys. Rev. Lett.* **82** 1971
[50] Sørensen A and Mølmer K 2000 Entanglement and quantum computation with ions in thermal motion *Phys. Rev.* A **62** 022311
[51] Qiskit Contributors 2023 Qiskit: an open-source framework for quantum computing (available at: https://doi.org/10.5281/ZENODO.2562111)
[52] Treiber A, Poppe A, Hentschel M, Ferrini D, Lorünser T, Querasser E, Matyus T, Hübel H and Zeilinger A 2009 A fully automated entanglement-based quantum cryptography system for telecom fiber networks *New J. Phys.* **11** 045013
[53] Krutyanskiy V, Canteri M, Meraner M, Bate J, Krcmarsky V, Schupp J, Sangouard N and Lanyon B P 2023 Telecom-wavelength quantum repeater node based on a trapped-ion processor *Phys. Rev. Lett.* **130** 213601
[54] Stephenson L 2019 Entanglement between nodes of a quantum network *PhD Thesis* University of Oxford
[55] Wang Y, Um M, Zhang J, An S, Lyu M, Zhang J-N, Duan L-M, Yum D and Kim K 2017 Single-qubit quantum memory exceeding ten-minute coherence time *Nat. Photon.* **11** 646–50
[56] Bock M, Eich P, Kucera S, Kreis M, Lenhard A, Becher C and Eschner J 2018 High-fidelity entanglement between a trapped ion and a telecom photon via quantum frequency conversion *Nat. Commun.* **9** 1998
[57] Schupp J, Krcmarsky V, Krutyanskiy V, Meraner M, Northup T E and Lanyon B P 2021 Interface between trapped-ion qubits and traveling photons with close-to-optimal efficiency *PRX Quantum* **2** 020331
[58] Krutyanskiy V, Meraner M, Schupp J, Krcmarsky V, Hainzer H and Lanyon B P 2019 Light-matter entanglement over 50 km of optical fibre *npj Quantum Inf.* **5** 72
[59] Stute A, Casabone B, Brandstätter B, Friebe K, Northup T E and Blatt R 2013 Quantum-state transfer from an ion to a photon *Nat. Photon.* **7** 219–22
[60] Casabone B, Friebe K, Brandstätter B, Schüppert K, Blatt R and Northup T E 2015 Enhanced quantum interface with collective ion-cavity coupling *Phys. Rev. Lett.* **114** 023602
[61] Stute A, Casabone B, Schindler P, Monz T, Schmidt P O, Brandstätter B, Northup T E and Blatt R 2012 Tunable ion–photon entanglement in an optical cavity *Nature* **485** 482–5
[62] Maurer U and Renner R 2016 From indifferentiability to constructive cryptography (and back) *Theory of Cryptography: 14th Int. Conf.* (Springer) pp 3–24

[63] van Dam J 2024 Data underlying the publication: hardware requirements for trapped-ion-based verifiable blind quantum computing with a measurement-only client (available at: https://data.4tu.nl/datasets/c98e631c-a6d3-411e-ad4c-4decf6943f57/1)
[64] Maier D, Rabbie J, Jirovska H, Knegjens R and Nijsten L 2023 Netsquid-ae snippet (available at: https://docs.netsquid.org/snippets/netsquid-ae/)
[65] Coopmans T, Dahlberg A, Skrzypczyk M and Wubben L 2023 Netsquid-nv snippet (available at: https://docs.netsquid.org/snippets/netsquid-nv/)
[66] Zwerger M, Lanyon B P, Northup T E, Muschik C A, Dür W and Sangouard N 2017 Quantum repeaters based on trapped ions with decoherence-free subspace encoding *Quantum Sci. Technol.* **2** 044001
[67] Jones R C 1941 A new calculus for the treatment of optical systems. I. Description and discussion of the calculus *J. Opt. Soc. Am.* **31** 488–93
[68] Theocaris P S and Gdoutos E E 1979 *Two-Dimensional Photoelasticity* (Springer) pp 113–31
[69] Lewis R 2015 *A Guide to Graph Colouring* vol 7 (Springer)
[70] Hoeffding W 1965 Asymptotically optimal tests for multinomial distributions *Ann. Math. Stat.* **36** 369–401
[71] Labay-Mora A, da Silva F F and Wehner S 2023 Reducing hardware requirements for entanglement distribution via joint hardware-protocol optimization (arXiv:2309.11448)
[72] da Silva F F, Avis G, Slater J A, and Wehner S 2023 Requirements for upgrading trusted nodes to a repeater chain over 900 km of optical fiber (arXiv:2303.03234)
[73] Masterov M, Torres A and Maier D 2023 YOTSE Github repository (available at: https://github.com/SURFQuantum/yotse/tree/main)
[74] Avis G, Coopmans T, Dahlberg A, Jirovská H, Maier D, Rabbie J and Skrzypczyk M 2023 Netsquid magic snippet (available at: https://docs.netsquid.org/snippets/netsquid-magic/)
[75] van Dam J 2023 Code underlying the publication: hardware requirements for trapped-ion-based verifiable blind quantum computing with a measurement-only client (available at: https://gitlab.tudelft.nl/janicedam/measurement-only-vbqc-code.git)