New Journal of Physics
The open-access journal for physics

Ancilla-based quantum simulation

Katherine L Brown[1,4,5], **Suvabrata De**[1], **Vivien M Kendon**[1] **and William J Munro**[2,3]

[1] Department of Physics and Astronomy, University of Leeds, Leeds LS2 9JT, UK
[2] NTT Basic Research Laboratories, NTT Corporation, 3-1 Morinosato-Wakamiya, Atsugi-shi, Kanagawa-ken 243-0198, Japan
[3] National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
E-mail: katherineb@lsu.edu

New Journal of Physics **13** (2011) 095007 (21pp)
Received 25 February 2011
Published 14 September 2011
Online at http://www.njp.org/
doi:10.1088/1367-2630/13/9/095007

Abstract. We consider the simulation of the Bardeen, Cooper and Schrieffer (BCS) Hamiltonian, a model of low-temperature superconductivity, on a quantum computer. In particular, we consider conducting the simulation on the qubus quantum computer, which uses a continuous variable ancilla to generate interactions between qubits. We demonstrate an $O(N^3)$ improvement over previous studies conducted on an NMR computer (Wu *et al* 2002 *Phys. Rev. Lett.* **89** 057904 and Brown *et al* 2006 *Phys. Rev. Lett.* **97** 050504) for the nearest-neighbour and completely general cases. We then proceed to show methods for minimizing the number of operations needed per time step using the qubus in three cases: the completely general case, the case of exponentially decaying interactions and the case of fixed range interactions. We make these results controlled on an ancilla qubit so that we can apply the phase estimation algorithm, and hence show that when $N \geqslant 5$, our qubus simulation requires significantly fewer operations than a similar simulation conducted on an NMR computer.

[4] Current address: School of Physics and Astronomy, Louisiana State University, Baton Rouge, LA 70808, USA.
[5] Author to whom any correspondence should be addressed.

Contents

1. Introduction

Quantum computing is a field of research that exploits the quantum behaviour of systems to gain advantages over standard classical digital computing. The most famous example is Shor's algorithm, which gives an exponential improvement in factoring numbers compared to the best-known classical algorithm [1]. However, even if we ignore the need for fault tolerance, such algorithms require thousands of qubits to obtain results inaccessible to classical computers and are therefore unlikely to be useful in the near future. A more accessible problem that is comparably hard to perform on a classical computer is simulating quantum systems. The idea of using quantum computers to simulate quantum systems was first proposed in 1982 by Feynman [2], and further developed by Lloyd [3] in 1996. Quantum simulation on a quantum computer would provide us with efficient algorithms to obtain: ground states of molecules [4]; details of bond formation [5]; and eigenvalues and eigenvectors [6] of many-body quantum Hamiltonians; for a recent review, see Brown *et al* [7]. One of the largest systems of qubits that has been simulated in full generality on a classical computer consists of only 36 qubits [8]. By taking advantage of the symmetries that occur in many quantum systems it is possible to simulate larger systems on a classical computer [9]. While 36 qubits is still a significant increase on what is currently available, it is an order of magnitude smaller than the break even point for many other quantum algorithms. Even with only tens of qubits it is possible to obtain interesting results that would be computationally intensive to obtain classically, especially when considering the general case scenarios. With more advanced and highly controlled quantum computers, it will be possible to get saving on important simulations such as chemical reactions; hence, not only is quantum simulation a testing ground for quantum computing, but also it will greatly increase our understanding of many scientific problems.

One particular system of interest for an early simulation is the Bardeen, Cooper and Schrieffer (BCS) Hamiltonian, a model of superconductivity formulated by Bardeen *et al* [10]. Superconductivity is still a poorly understood phenomenon, and the BCS model is currently

one of the best descriptions available. The BCS Hamiltonian is also used to describe pairing in nuclear physics, where it is called the pairing force Hamiltonian [11]. Pairing Hamiltonians are used to describe many processes in condensed matter physics and therefore a technique for simulating the BCS Hamiltonian should be adaptable to many other purposes. The BCS Hamiltonian, H_{BCS}, is

$$H_{\mathrm{BCS}} = \sum_{m=1}^{N} \frac{\epsilon_m}{2}(n_m^F + n_{-m}^F) + \sum_{m,l=1}^{N} V_{ml} C_m^\dagger C_{-m}^\dagger C_{-l} C_l, \tag{1}$$

where $C_m^\dagger$ and C_m are the Fermionic creation and annihilation operators, $n_{\pm m}^F = C_{\pm m}^\dagger C_{\pm m}$ are the number operators, N is an effective state number that, represents the number of modes to be simulated, V_{ml} is the interaction potential and ϵ_m is the on-site energy of a pair in mode m. Pairs of fermions have quantum numbers m and $-m$ where pairs have equal energy but opposite momentum and spin, $m = (\mathbf{p}, \uparrow)$ and $-m = (-\mathbf{p}, \downarrow)$.

The energy gap between the ground and first excited states of the BCS Hamiltonian is a non-perturbative function; this means that the Hamiltonian needs to be exactly diagonalized to obtain an accurate value for the energy gap. The non-pertubative nature of the Hamiltonian also means that to get an accurate value for the energy gap it is important to consider information from all interactions near the Fermi surface including long-range interactions [12]. To obtain a solution for the energy gap, Bardeen *et al* [10] took $V_{ml} = -V$ for states near the Fermi surface and 0 elsewhere. This approximation works for most metallic superconductors, but there are some cases where it is inaccurate. In these cases, it would be useful to perform a simulation with generalized V_{ml}. Systems where this is useful include Fermi gases [13] and nuclear systems [14]. Anderson also looked at some cases where this might be beneficial [12], while Mastellone *et al* [15] looked at systems where the BCS ansatz might not hold. While there have been attempts to do this general simulation classically, many classical simulation methods have their limitations. Volya *et al* [16] provide an exact solution but require a low-dimensional Hilbert space. Dukelsky *et al* [17] require an approximation. Mastellone *et al* [15] are limited by the size of the possible diagonalization and the accuracy of the results. In the case of integrable systems, an exact solution can be obtained using the Richardson solution [11], which was originally developed for nuclear systems. However, it is worth noting that the resultant integrals can only be solved numerically in a limited number of cases. Very recent results obtained by Ho *et al* [18] obtain the energy gap of the BCS Hamiltonian without the need for the BCS approximation. However, these results are not as accurate for eigenvectors in the weak interaction regime. This means that to obtain an accurate result, direct diagonalization is often still necessary. However, it is resource intensive on a classical computer to diagonalize the exact Hamiltonians for more than a few tens of qubits. Therefore, it is still difficult to characterize systems for cases where V_{ml} cannot be held constant. Wu *et al* [19] have shown that on a quantum computer it is possible to conduct a simulation of the BCS Hamiltonian efficiently and then use this simulation to extract information about the energy gap. While they concentrate on a nearest-neighbour case, they also describe a case with general V_{ml}. Quantum simulation should thus be able to confirm whether the BCS Hamiltonian is correct for the metallic superconductors not covered by the assumption $V_{ml} = -V$.

To conduct a simulation of the BCS Hamiltonian on a quantum computer, we need to map equation (1) to the qubit Hamiltonian. This mapping is worked out in [19] and results in the

BCS Hamiltonian taking the form

$$H_{\text{pair}} = \sum_{m=1}^{N} \frac{\varepsilon_m}{2}\sigma_{zm} + \sum_{m<l}^{N} \frac{V_{ml}}{2}(\sigma_{xm}\sigma_{xl} + r\sigma_{ym}\sigma_{yl}), \tag{2}$$

where $\sigma_{xk}/\sigma_{yk}/\sigma_{zk}$ are the Pauli $X/Y/Z$ on the kth qubit, and $\varepsilon_m = \epsilon_m + V_{mm}$ and r is a parameter determined by the mapping.

As we are interested in determining the energy gap between the ground and first excited states, we start by preparing our system in a superposition of these two states. This initial state is prepared using the technique proposed by Wu *et al* [19] for preparing an initial state on an NMR computer. We first prepare our system of N qubits in a state of all zeros; this is a basic ground state. We then apply spin flips so we have a basis state containing n ones, where n is the number of excitations in the BCS Hamiltonian. This is different from N, which is the number of modes in the BCS Hamiltonian and the number of qubits in our quantum computer. Using a quasi-adiabatic method of applying our Hamiltonian, we prepare a superposition of the ground and first excited states; we discuss this step in more detail in section 4. This generates an initial state which will allow us to extract data about the low-lying energy spectrum of our Hamiltonian and thus determine the energy gap.

To perform the time evolution and extract the data, we use the phase estimation algorithm [20]; this involves performing the BCS Hamiltonian controlled on a set of ancilla qubits for a set of time intervals given by $2^{k-1}t$ for the kth ancilla qubit. This encodes the data about the time evolution of the BCS Hamiltonian into the ancilla qubits. A quantum Fourier transform (QFT) is then used to extract data from the system. In section 5, we look at the number of operations required to implement our unitaries in a controlled fashion, and in section 6 we consider the number of operations needed to perform our QFT.

We will consider conducting our simulation on an ancilla-driven quantum computer, where an auxiliary system is used to generate interactions in the main processing unit. The main advantage of such a system is that it allows us to generate entanglement between distant qubits without the need for swap gates. In terms of the BCS Hamiltonian, this facilitates the simulation of long-range interactions, something difficult to do classically. While our results are applicable to ancilla-driven qubit computers in general, we will concentrate in particular on the qubus quantum computer. The qubus is a continuous variable ancilla that is used to generate interactions between the qubits [21, 22]. The continuous variable bus is advantageous because detecting and controlling single photons is experimentally difficult, whereas the continuous variable is easier to control. The qubus system generates gates between qubits deterministically without the need for measurement, but does require us to be able to generate an interaction between the field and a matter qubit [22].

In this paper, we describe a method for simulating the time evolution of the BCS Hamiltonian and then extracting the energy gap, using a similar technique to the one proposed by Wu *et al* [19]. We show that the use of a qubus quantum computer gives significant advantages over an NMR quantum computer, and we proceed to show how one can obtain significant improvements over a naïve qubus implementation. This allows us to obtain information about the ground and first excited states of the Hamiltonian more efficiently than Wu *et al* [19]. The simulation is limited in accuracy only by the number of terms used in the Trotter approximation to combine the non-commuting parts of the Hamiltonian, and the number of ancilla qubits. In both cases, a linear increase in precision costs a linear increase in resources [23]. Other than this, no approximations are used and we can obtain results in

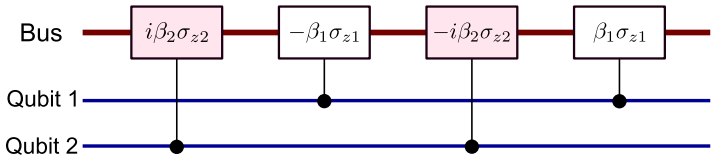

Figure 1. A circuit diagram for implementing the C-Phase gate on the qubus. The boxes show displacements performed on the continuous variable field controlled by the qubits as in equation (3). Shaded boxes represent an operation acting on the position quadrature of the bus, whereas unshaded boxes represent operations on the momentum quadrature.

the completely general case where the coupling between pairs can take any value for any pair. We can therefore access regimes unobtainable using classical approximation methods.

The paper is organized as follows. In section 2, we introduce the qubus quantum computer and discuss how we go about reducing the number of interactions with the bus needed to perform certain operations. Section 3 contains three methods for producing the necessary unitaries for the BCS Hamiltonian on the qubus; section 3.1 concentrates on the general case; section 3.2 looks at the case where it is possible to obtain an order of N saving in return for a reduction in generality; and section 3.3 looks at fixed range interactions. Section 7 puts the results of sections 4–6 together to work out how many operations are needed as part of the longest single run of the qubus computer. We summarize our results in section 8.

2. The qubus quantum computer

A qubus quantum computer is a hybrid system consisting of a processing unit made of qubits and a continuous variable field 'bus' that generates interactions and transfers information between the qubits. Using the bus to generate interactions between distant qubits removes the need to either change calibration settings every time a new qubit is added or use swap operations to move qubits next to each other. There are several proposals for physical architectures that could potentially be used to build a qubus system. These include optical quantum systems [21] and superconducting systems [24].

Interactions in the qubus architecture take the form of displacement operators applied to the continuous variable field. These can be written in the form [22]

$$D(\beta\sigma_{zk}) = \exp(\beta\sigma_{zk}a^\dagger - \beta^*\sigma_{zk}a), \tag{3}$$

where $a^\dagger$ and a are the field creation and annihilation operators, σ_{zk} is the Pauli Z operator acting on the kth qubit, $\beta = \chi t\, \mathrm{e}^{\mathrm{i}(\phi-\frac{\pi}{2})}$, and χ is the strength of the nonlinearity being used, which is chosen to be real. In the qubus system, we use $\phi = 0$, corresponding to the position quadrature, or $\phi = \frac{\pi}{2}$, corresponding to the momentum quadrature.

The displacement operators entangle the bus with the qubits. A qubit is entangled to either the position or momentum quadrature of the bus, and since these displacements are in orthogonal directions it is possible to create a maximally entangling gate. Using two qubits and one bus it is possible to generate a gate which is locally equivalent to a controlled phase (C-Phase) gate by performing just four displacement operations on the bus. This scheme is illustrated in figure 1 and is given by the operator $D(\beta_1\sigma_{z1})D(-\mathrm{i}\beta_2\sigma_{z2})D(-\beta_1\sigma_{z1})D(\mathrm{i}\beta_2\sigma_{z2})$; see [22].

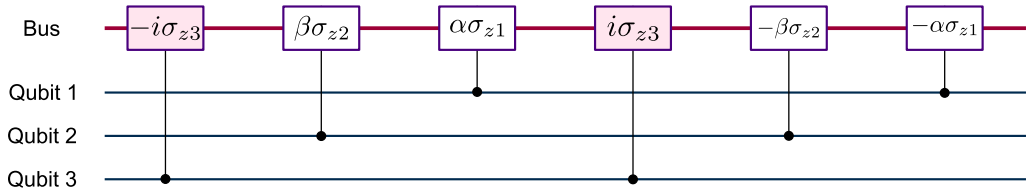

Figure 2. A reduced gate sequence for $\exp(\mathrm{i}\alpha\sigma_{z1}\sigma_{z3} + \mathrm{i}\beta\sigma_{z2}\sigma_{z3})$ using six operations instead of eight. The boxes show displacements performed on the continuous variable field controlled by the qubits. Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.

These operations are combined using the formula

$$D(a)D(b) = \exp\left(\frac{ab^* - a^*b}{2}\right) D(a+b), \tag{4}$$

where a and b are Pauli operators on the qubits and must commute (e.g. because they act on different qubits). This sequence thus results in the interaction $\exp(2\,\mathrm{i}\beta_1\beta_2\sigma_{z1}\sigma_{z2})$, which when $2\beta_1\beta_2 = \pi/4$ is equivalent under local unitaries to a C-Phase gate.

We will define the 'naïve' method of implementing an operation on a qubus as implementing a Hamiltonian by performing each C-Phase gate individually, disentangling the bus from the system between each set of gates. This technique requires four operations for each C-Phase gate within the desired interaction. If we take an interaction of the form $\exp(\mathrm{i}\,\alpha\sigma_{z1}\sigma_{z3} + \mathrm{i}\,\beta\sigma_{z2}\sigma_{z3})$, then using the naïve method would require a total of eight operations to implement it. However, the term σ_{z3} appears twice; therefore it is possible to reduce the total number of bus operations by rearranging the gate sequence, so the third qubit only needs to interact with the bus twice. The gate sequence required to do this is shown in figure 2; it uses only six operations.

Provided there are 'overlapping' terms such as the σ_{z3} in the above example, it is possible to get a reduction in the number of operations required over the naïve method. The extent of this reduction depends upon the number of terms that overlap, how often these overlapping terms appear and the level of generality required. We will now apply these techniques to reduce the number of operations needed to simulate the BCS Hamiltonian.

3. Implementing the Hamiltonian

We will now consider how to perform the BCS Hamiltonian on our qubus quantum computer; this stage is essential for both the initialization process, where it is used for a semi-adiabatic evolution, and the time-evolution stage. The time evolution of the BCS Hamiltonian in terms of Pauli operators is shown in equation (2), which consists of three non-commuting Hamiltonians so that $H_{\mathrm{pair}} = H_0 + H_{xx} + H_{yy}$.

Taking the corresponding unitaries to these Hamiltonians, we obtain

$$U_0 = \exp\left(\mathrm{i}\sum_m^N \frac{\varepsilon}{2}\sigma_{zm}\right), \tag{5}$$

$$U_{xx} = \exp\left(\mathrm{i}\sum_{m<l}^{N}\frac{V_{ml}}{2}(\sigma_{xm}\sigma_{xl})\right), \tag{6}$$

$$U_{yy} = \exp\left(\mathrm{i}\sum_{m<l}^{N}\frac{V_{ml}}{2}r(\sigma_{ym}\sigma_{yl})\right). \tag{7}$$

These unitaries do not commute; therefore $\exp(H_{\mathrm{pair}}) \neq \exp(H_0)\exp(H_{xx})\exp(H_{yy})$. To get around this, we use the Trotter approximation [25, 26] splitting the time t into small segments τ. In the first order case, this is given by $\exp(H_{\mathrm{pair}}) \approx \exp(H_0\tau)\exp(H_{xx}\tau)\exp(H_{yy}\tau) + \mathrm{O}(\tau^2)$, and in the second order case, it is given by $\exp(H_{\mathrm{pair}}) \approx \exp(H_0\tau/2)\exp(H_{xx}\tau/2)\exp(H_{yy}\tau)\exp(H_{xx}\tau/2)\exp(H_0\tau/2) + O(\tau^3)$. Longer time intervals are built up by performing the short time evolution multiple times. While at first it may seem possible to implement the entire Hamiltonian without using the Trotter approximation, this is not possible because the necessary displacement operators are based on Pauli operators, which do not commute. We can see that this is the case because equation (4) is only valid for commuting operators.

In equation (5), U_0 is a sum of local operations and can be implemented using N local unitaries. We perform these local unitaries by assuming that we can address each qubit individually, thus perform the required operation without bus operations. We outline how to perform a general local unitary using just a single local unitary, $\exp(\mathrm{i}\pi\sigma_x/4)$, its inverse and bus operations in [27]. However, for simplicity and to reduce the total number of operations required, we assume that we can perform any local unitary directly, in this instance. In equations (6) and (7), U_{xx} and U_{yy} are equivalent under local unitaries to

$$U_{zz} = \exp\left(\mathrm{i}\sum_{m<l}^{N}\frac{V_{ml}}{2}(\sigma_{zm}\sigma_{zl})\right). \tag{8}$$

We now consider how to implement equation (8) in three different cases.

3.1. The fully generalized case

Equation (8) is a sum of $(N^2-N)/2$ terms, each requiring one gate of the form $\exp(\mathrm{i}\,\beta_m\beta_l\sigma_{zm}\sigma_{zl})$. For the simplest possible implementation on a qubus computer, we perform each of these gates individually requiring four bus operations per gate. This means that U_{zz} requires a total of $2N^2-2N$ bus operations to perform.

This technique uses the simplest possible procedure for generating the necessary interactions, and it should be possible to get reductions by reusing our bus. However, if we wish to be able to set each of our constants, V_{ml}, separately, then we have significant constraints on how we can do this. A simple lower bound on the number of bus operations needed can be found by considering the number of individual constants. For a system being modelled on N qubits, we require $(N^2-N)/2$ constants, which have no dependence upon each other. Since our operations on the bus occur in pairs, and each pair can only produce one constant, we can see that it would be impossible to generate this using less than N^2-N bus operations.

We now consider how we would go about generating the necessary bus operations. We want qubit 1 to interact with all the other $N-1$ qubits, with completely independent constants. The simplest way to do this would be to connect qubit 1 to one quadrature of the bus and

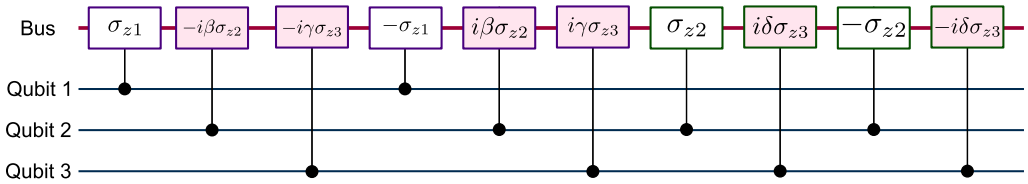

Figure 3. A circuit for simulating U_{zz} for three qubits with generalized constants V_{ml} in ten operations. The boxes show displacements performed on the continuous variable field controlled by the qubits. Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.

then connect the other $N-1$ qubits to the other quadrature of the bus. Now, when we consider qubit 2, we can see that we have already generated the interaction between this and qubit 1; therefore it only needs to interact with the remaining $N-2$ qubits. However, we want the constants generated from these interactions to be completely independent of the ones generated by interacting qubit 1 with the others. If we hold too many qubits on the bus, this independence is not possible; therefore, after each step, we need to disconnect all of the qubits from both quadratures of the bus.

We therefore consider an $N-1$ step process, where for the ath step we connect qubit a to one quadrature of the bus, and the other $a-1$ qubits to the other quadrature before disconnecting qubit a and then all the others. The total number of bus operations needed for this is given by

$$\sum_{a=2}^{N} 2a = 2\left(\sum_{a=1}^{N} a\right) - 2 = N^2 + N - 2\,. \tag{9}$$

A circuit for generating interactions between three qubits using this technique is shown in figure 3.

One way of reducing the number of operations further is to leave some qubits on the bus at the end of each step. We now consider the case where we first attach qubit 1 to the first quadrature of the bus and the other $N-1$ qubits to the second quadrature, as before. However, when we come to disentangling the qubits, we leave qubit 2 entangled to the second quadrature of the bus. Step 2 then involves attaching qubits 3 through N to the first quadrature of the bus, thus generating entanglement between all these qubits and qubit 2. In the disentangling step, all the qubits are removed from the bus except qubit 3, which is used for step 3. In this scenario, qubits 1 and 2 interact with the bus twice (one connect and one disconnect) each, qubit 3 interacts with the bus four times (two connects and two disconnects), etc. This occurs all the way up to qubit N, which interacts with the bus $2(N-1)$ times. This leads to a total number of operations given by

$$2\left(\sum_{a=1}^{N-1} a\right) + 2 = N^2 - N + 2\,. \tag{10}$$

This is $2N-4$ operations less than our previous result and just two more operators than our lower bound. A circuit for performing this technique using three qubits is shown in figure 4. It is less obvious that in this case the constants can be completely general as one constant from

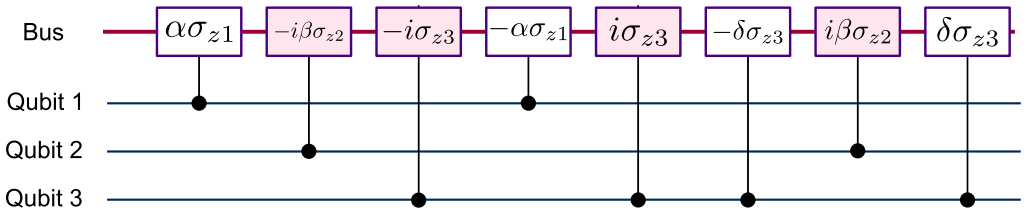

Figure 4. A circuit for simulating U_{zz} for three qubits with generalized constants V_{ml} in eight operations. The boxes show displacements performed on the continuous variable field controlled by the qubits. Shaded boxes represent an operation acting on the position quadrature of the bus, whereas unshaded boxes represent operations on the momentum quadrature.

each step will carry through to the next step. However, since all the constants in the next step can be set arbitrarily it is possible to achieve this. If there is no interaction between two qubits, e.g. between qubits 2 and 3, then the higher numbered qubit is skipped in that step and added in during a later step. This will never require extra operations and will in some cases provide a saving.

3.2. A limited case

While the completely general case is important, many physical systems do not require this level of generality. If we are prepared to restrict our constants so that they become dependent upon each other, then it is possible to get saving of $O(N)$ over our previous techniques. This provides a saving for a range of different scenarios including ones where the strength of the interaction drops off exponentially with distance. We will begin by considering the simplest possible case of wanting to create the U_{xx} and U_{yy} that are local unitaries away from U_{zz} in equation (8) when V has no dependence upon either m or l. The minimal number of bus operations needed to generate this can be worked out by noting that for the sum given, both m and l can take $N-1$ possible different values. This means that $2(N-1)$ bus operations are needed to cover all the possible values of m and similarly for l. This results in a total number of bus operations given by $4(N-1)$.

A method of performing the operations that meets this bound involves attaching qubits 2 through N to the first quadrature of the bus. Qubit 1 is then attached to the second quadrature, generating entanglement between it and all the other qubits. So far, this matches the method in section 3.1. However, now, instead of removing all the qubits from the bus, we disconnect qubit 2 from the first quadrature of the bus and then reattach it to the second quadrature. This generates entanglement between it and qubits 3 through N. We continue this process, removing qubits 3 through $N-1$ from the first quadrature and then attaching them to the second quadrature. Finally, when we come to removing qubit N from the bus, we do not reattach it to the second quadrature. This leaves the first quadrature of the bus completely empty and $N-1$ qubits entangled to the second quadrature of the bus. The final step involves removing all the remaining qubits from the second quadrature of the bus. This means that every qubit interacts with the bus four times (two connections and two disconnections), apart from the first and last qubits, which only interact with the bus twice. The total number of bus operations needed is given by $4N-4$; therefore this technique meets the lower bound exactly. A circuit showing the sequence of bus operations for this technique in the four-qubit case is shown in figure 5.

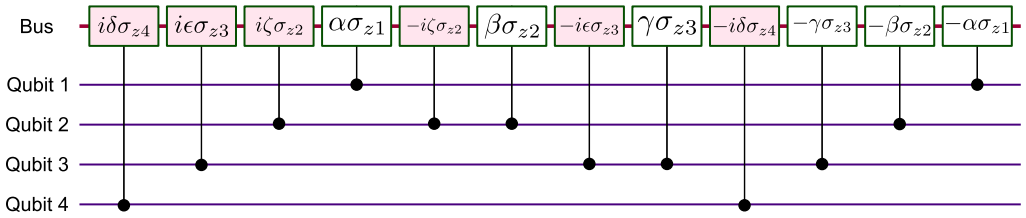

Figure 5. A circuit for simulating U_{zz} with a strong dependence between the constants. The boxes show displacements performed on the continuous variable field controlled by the qubits. Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.

This technique can give constants more general than a fixed V. The constants attached to each qubit give a matrix of values for V that are dependent upon m and l and are linked as shown below:

$$\begin{array}{cccc} & 2 & 3 & 4 \\ 1 & \alpha\zeta & \alpha\epsilon & \alpha\delta \\ 2 & & \beta\epsilon & \beta\delta \\ 3 & & & \gamma\delta. \end{array}$$

The entries of the matrix above represent the strength of the interaction between the numbered qubits given the constants in figure 5; so, for example, the second column of the first row represents the strength of the interaction between qubits 1 and 3. By looking at the matrix we can see that it would not be possible to set values for α, β, γ, δ, ϵ and ζ such that all six entries of the matrix could be set completely arbitrarily. We can see this easily because $\alpha\epsilon/\alpha\delta$ has to be equal to $\beta\epsilon/\beta\delta$ if α, β, δ and ϵ are complex constants; however, if we set all six entries of our matrix randomly, then this is not always the case.

This dependence allows numerous cases, including an exponential fall-off with distance or other more complicated dependences. One example of exponential decay would be setting $\alpha = 1$, $\beta = 2$, $\gamma = 3$, $\delta = 1/4$, $\epsilon = 1/2$ and $\zeta = 1$. The possible constants provide an interesting set of cases relevant to realistic situations.

3.3. Fixed range interactions

One case not covered by our limited case, in which it is obviously possible to get improvements, is nearest-neighbour interactions. This would consist of a chain of qubits and is explored by Louis *et al* [28] in the context of building cluster states. This involves a U_{zz} of the form

$$U_{zz} = \exp\left(\sum_{m=1}^{N-1} \frac{V_{m,m+1}}{2} \left(\sigma_{zm}\sigma_{z(m+1)}\right)\right). \tag{11}$$

Louis *et al* found that for a chain of N qubits, it is possible to generate nearest-neighbour interactions using just $2N$ bus operations. A circuit illustrating a suitable scheme is shown in figure 6. This can be expanded to include a U_{zz} with both nearest-neighbour and next-to-nearest-neighbour interactions, a scenario that would require $4N - 4$ bus operations.

IOP Institute of Physics DEUTSCHE PHYSIKALISCHE GESELLSCHAFT

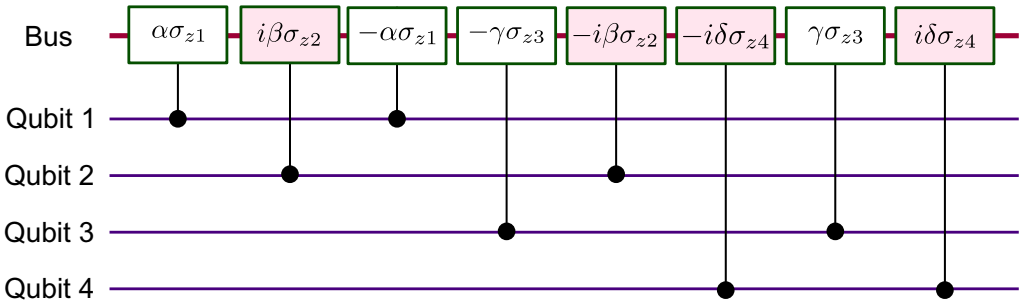

Figure 6. A circuit for simulating U_{zz} with nearest-neighbour interactions. The boxes show displacements performed on the continuous variable field controlled by the qubits. Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.

It is possible to derive a general result for fixed range interactions by introducing a new variable p, where p represents the range of the interaction. When $p = 1$, we consider only nearest-neighbour interactions; when $p = 2$, we consider nearest-neighbour and next-to-nearest-neighbour, etc. We split our terms into a cycle of several steps, where we leave one qubit active with the bus at the end of each step; this allows us to keep the interaction strengths between various qubits completely generalized. For the first step, $p+1$ qubits need to interact with the bus twice; this represents qubit 1 and the p qubits connected to it. The next set of steps need p qubits to interact with the bus twice; since we assume that the first qubit in each step is already connected to the bus, we only need to attach its p-nearest neighbours. We require $2p$ qubits operation on the bus for every single step except for the very first one and the p final steps. We therefore need to perform $2p$ operations a total of $N - p - 1$ times. Finally, for the last set of steps there are no longer p qubits left in the chain connected to our active qubit; this means that the number of bus operations required decreases from $2(p-1)$ to 0 in intervals of 2. Therefore, for an interaction length p, the number of operations on the bus needed to generate U_{zz} is given by

$$2(p+1) + 2p(N-p-1) + 2\sum_{a=1}^{p}(p-a) = 2pN - p^2 - p + 2\,. \tag{12}$$

We can check that this equation agrees with our previous results. If we take the nearest-neighbour case, then $p = 1$ and we find that the total number of operations needed to generate U_{zz} is $2N$. If we take the case where all the qubits interact with each other, $p = N - 1$; therefore $N^2 - N + 2$ bus operations are needed to generate U_{zz}. This demonstrates a technique that allows the generation of interactions with a cut-off at an arbitrary distance. It should be possible to obtain further saving in these cases using a technique similar to that described in section 3.2.

4. Initialization

To initialize our system we need to generate a superposition of the ground and first excited states for our BCS Hamiltonian. To do this, we can use the same method as Wu *et al* [19] for the initialization procedure within an NMR system. We begin by preparing our system so that

we have N qubits all in state $|0\rangle$, where N is the number of modes in the BCS Hamiltonian; this is a basic ground state. By applying spin flips it is possible to transform this state so that n qubits are in state $|1\rangle$, where n is the number of excitations within our BCS Hamiltonian. This generates a computational basis state and the process is trivial to perform on a qubus quantum computer.

To get our qubus quantum computer to be in a combination of the ground and first excited states of the BCS Hamiltonian, we need to implement an adiabatic form of our time evolution given by

$$U_{ad}(S\tau) \approx \mathrm{e}^{-H(k\tau)\tau} \cdots \mathrm{e}^{-\mathrm{i}H(2\tau)\tau}\mathrm{e}^{-H(\tau)\tau} + O(\tau^2), \tag{13}$$

where $\exp[-\mathrm{i}H(j\tau)\tau] \approx \exp(\mathrm{i}H_0\tau)\exp[-\mathrm{i}c(j\tau)H_1\tau]$, $j = 1, \ldots, S$, $S\tau = T$ and c is a function that varies slowly with time, t; c goes from 0 at $t = 0$ to 1 at $t = T$. In this case, $H_1 = H_{xx} + H_{yy}$. We are gradually increasing the strength of the interactions within the Hamiltonian so that we go from a basis state of H_0 to a state that is described by the BCS Hamiltonian. If we conducted this evolution in accordance with the adiabatic condition ($S \gg \pi/(\tau\Delta)$, where 2Δ is the energy gap between the ground and first excited states), then we would end up in the ground state of the BCS Hamiltonian. However, instead we relax the adiabatic condition and, as a result, end up with a component of the first excited state mixed in with the ground state. In the case of the BCS Hamiltonian, the short time approximation is valid when $\tau \ll 1/d$ where d is the level spacing and τ the length of a single time interval. This gives $S \gg \pi d/\Delta$. Brown *et al* [23] add in another factor which represents the precision of the desired energy gap, and is given by δ. This results in the condition $S = \pi d/\delta\Delta$. Taking $d/\Delta = 0.1$ from Wu *et al* [19], the total number of time steps required is given by

$$S = 0.1\pi/\delta. \tag{14}$$

The time evolution here can be implemented using the method described in section 3. As we are considering a short evolution we can consider using only the first-order Trotter approximation, similar to Wu *et al* [19]. This means H_0 can be implemented using single local unitaries on each qubit and H_1 will require twice the number of operations needed to perform H_{zz} plus $4N$ local unitaries to transform our U_{zz} terms to U_{xx} and U_{yy}. We can therefore see that the number of operations per time step, $I(N)$, in the general case will be

$$I_{\mathrm{G}}(N) = 2N^2 + 3N + 4. \tag{15}$$

In the limited case, we have

$$I(N) = 13N - 8, \tag{16}$$

whereas in the case of fixed range interactions, we obtain

$$I_{\mathrm{L}}(N) = 4pN + 5N - 2p^2 - 2p + 4. \tag{17}$$

To work out the total number of operations needed for the initialization procedure, we multiply $I(N)$ by S. There are potential problems with this adiabatic evolution step, particularly if there is an exponentially small-energy gap between the ground and first excited states [7]. However, it is hoped that this procedure should be valid for certain physical Hamiltonians, and as we are performing a quasi-adiabatic evolution rather than an adiabatic evolution, components of the excited states are desired, rather than a problem.

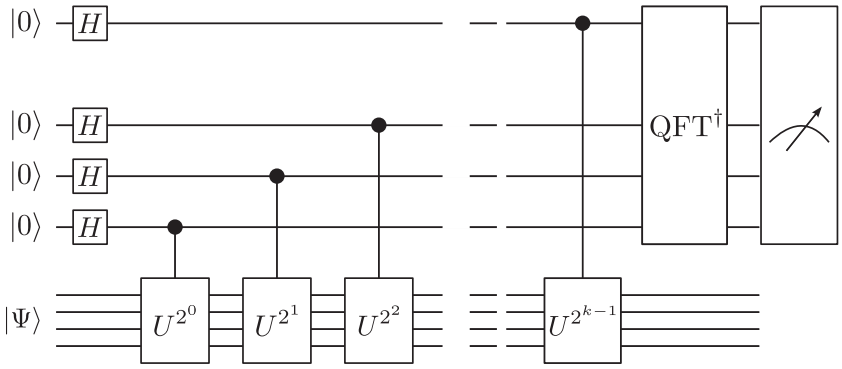

Figure 7. A circuit for performing the phase estimation algorithm; the sequence of controlled unitaries encodes phase information from the input state $|\Psi\rangle$ in the ancilla, which is then extracted using the QFT.

5. The phase estimation algorithm

Once we have created the superposition of the ground and excited states, we want to use the phase estimation algorithm for our data extraction procedure. The phase estimation algorithm has two stages; the first stage implements the unitary representing the BCS Hamiltonian and therefore encodes the data about our evolution into a series of ancilla qubit; the second uses a QFT to extract the data.

A circuit for performing the phase estimation algorithm is shown in figure 7. To illustrate the workings of the phase estimation algorithm, we will consider our first register to be in a single eigenstate, $|u\rangle$, of U; therefore implementing U will leave $|u\rangle$ unchanged. Implementing CU^{2^j}, i.e. controlled on an ancilla qubit, results in

$$CU^{2j}\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)\,|u\rangle=\frac{1}{\sqrt{(2)}}\left(|0\rangle+\mathrm{e}^{\mathrm{i}2^j\Phi}|1\rangle\right)|u\rangle, \tag{18}$$

where $\mathrm{e}^{\mathrm{i}\Phi}$ given by $U|u\rangle=\mathrm{e}^{\mathrm{i}\Phi}|u\rangle$ is the eigenvalue of $|u\rangle$. We now consider applying these gates from $j=0$ to $j=k-1$ using k-ancilla qubits, with the net result

$$\frac{1}{\sqrt{2^k}}(|0\rangle+\mathrm{e}^{\mathrm{i}2^{k-1}\Phi}|1\rangle)\cdots(|0\rangle+\mathrm{e}^{\mathrm{i}2\Phi}|1\rangle)(|0\rangle+\mathrm{e}^{\mathrm{i}\Phi}|1\rangle)|u\rangle=\frac{1}{\sqrt{2^k}}\sum_{y=0}^{2^k-1}\mathrm{e}^{\mathrm{i}\Phi y}|y\rangle|u\rangle. \tag{19}$$

Tracing out the register containing the eigenstate leaves us with our k-ancilla qubits in the state $\sum_{y=0}^{2^k-1}\mathrm{e}^{\mathrm{i}\Phi y}|y\rangle$. Applying the inverse QFT encodes the phase into the register of ancilla qubits [29].

Assuming our register starts in an exact eignestate, the final result of the phase estimation procedure is $|\tilde{\Phi}\rangle|u\rangle$, where $|\tilde{\Phi}\rangle$ is a k-qubit approximation of $|\Phi\rangle$. If the first register starts in an approximation of the eigenstate, then the phase estimation algorithm results in the second register being transformed into $|\Phi\rangle$ with a probability that increases with the accuracy of the eigenstate and with k [29]. Similarly, if the register starts in a superposition of eigenstates, then the ancilla register ends up being in a superposition of the resultant eigenvalues in a ratio that depends on how much the corresponding eigenstate contributes to the initial input state. In our case, since we start in a superposition of the ground and the first excited state, we can extract

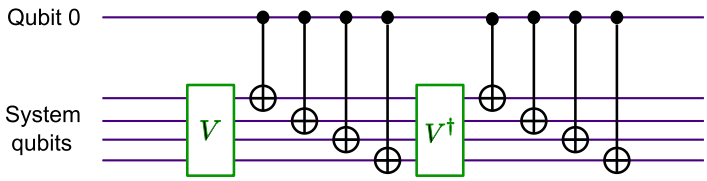

Figure 8. A sequence for making N local unitaries controlled on an ancilla qubit, where $V^2 = U_0 = U_{L1} \otimes \cdots \otimes U_{LN}$.

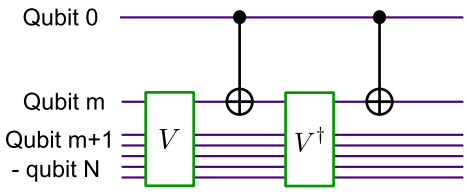

Figure 9. A circuit for making a unitary U controlled on an ancilla qubit. In this case, $U = V^2$, and U consists of pairs of Pauli Z operations where each pair of operations acts on qubit m.

the eigenvalue for the ground state and the eigenvalue for the first excited state by using several runs of the computation. This will allow us to work out the energy gap.

To implement our unitary as part of the phase estimation algorithm involves performing our operations in a controlled fashion, dependent upon the state of an ancilla qubit. As part of this we need to make our two-qubit operations and our single-qubit operations controlled on an ancilla qubit. It is possible to make a single-qubit unitary controlled on an ancilla qubit using the procedure outlined by Barenco *et al* [30]. Since our local unitaries in U_0 commute, it is possible to perform a set of gates that make our entire U_0 controlled on an ancilla qubit using the sequence of operations shown in figure 8. In this case, we have $V^2 = U_0$. This is more efficient than performing the gates individually because our CNOT sequence can be performed in $4N + 3$ operations, rather than the $7N$ required to perform N CNOT gates individually. The $4N + 3$ operations can be broken down into $2N + 2$ to perform the entangling operations with the bus, $2N$ to change our C-Phase gates into CNOT gates, and 1 to correct the ancilla qubit, to ensure that the necessary gate is performed. As part of the circuit in figure 8, the local unitaries need to be implemented twice, as does the sequence of CNOT gates; therefore the total number of operations is given by $10N + 6$.

We also need to consider how to implement our two-qubit unitaries in a controlled fashion. We consider making the term U_{zz} controlled and then transforming this state into U_{xx} or U_{yy} as discussed previously. We consider splitting our term U_{zz} into several terms, each one given by

$$U_{zz}(m) = \sum_{l=m+1}^{N} \frac{V_{ml}}{2} \sigma_{zm} \sigma_{zl} \tag{20}$$

and then making each term controlled independently. To do this, we use the circuit shown in figure 9.

If we consider qubit m from equation (20) to be the first qubit in our matrix, then the matrix representation of V, where $V^2 = U_{zz}(m)$, takes the form

$$W(m) = \begin{pmatrix} \exp(\mathrm{i}\alpha) & 0 \\ 0 & \exp(-\mathrm{i}\alpha) \end{pmatrix}. \tag{21}$$

We note than when $m = k$, the terms $\exp(\pm\mathrm{i}\alpha)$ act on qubits $k+1$ through N. We choose that representation because it is directly analogous to the one shown by Barenco *et al* [30], who show how to make a single-qubit unitary of this form controlled on an ancilla qubit. The circuit shown in figure 9 is identical, apart from the fact that W acts on multiple qubits. A proof that this works is outlined in [27]. The total number of operations for this procedure is given by

$$\sum_{a=2}^{N}(4a+16) = 2N^2 + 18N - 20. \tag{22}$$

In each cycle, $4a$ gates are required to perform V and $V^\dagger$, whereas 16 are needed for the two CNOT gates.

When combining our unitaries U_{xx}, U_{yy} and U_0, we use the Trotter approximation. Since we need a more accurate U than for initialization, we will use the second-order Trotter approximation [25, 26], which is given by

$$U_{\mathrm{BCS}} \approx [U_0(\tau/2)U_{xx}(\tau/2)U_{yy}(\tau)U_{xx}(\tau/2)U_0(\tau/2)], \tag{23}$$

where τ is the length of a single time interval. Figure 7 shows a circuit for performing the phase estimation algorithm. We will choose U to be an implementation of the BCS Hamiltonian for a single time interval in the Trotter approximation, U^2 will require two time intervals, U^4 will need four intervals, etc. This will allow us to work out the number of operations required as a function of the number of ancilla qubits. We therefore find that the total number of operations required for running the local unitaries in the phase estimation algorithm with k ancillas is given by $(2^k - 1)$ times the number of operations needed to implement U; therefore, we have

$$P_{\mathrm{G}}(N) = (2^k - 1)(6N^2 + 80N - 48) \tag{24}$$

in the completely general case, and

$$P_{\mathrm{L}}(N) = (2^k - 1)(12Np - 6p^2 - 6p + 86N - 48) \tag{25}$$

in the case of limited range interactions. The number of ancillas required is chosen dependent upon the desired accuracy.

6. Data extraction

As well as performing our controlled unitaries, the phase estimation algorithm requires us to implement the QFT; so we provide an efficient technique for performing the QFT on the qubus quantum computer. This provides us with significant saving over a naïve qubus implementation.

The QFT involves performing controlled rotation operators on the qubits in a form shown in figure 10, where the gate CR_a corresponds to a controlled phase gate of the form

$$CR_a = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mathrm{e}^{2\pi\mathrm{i}/2^a} \end{pmatrix}. \tag{26}$$

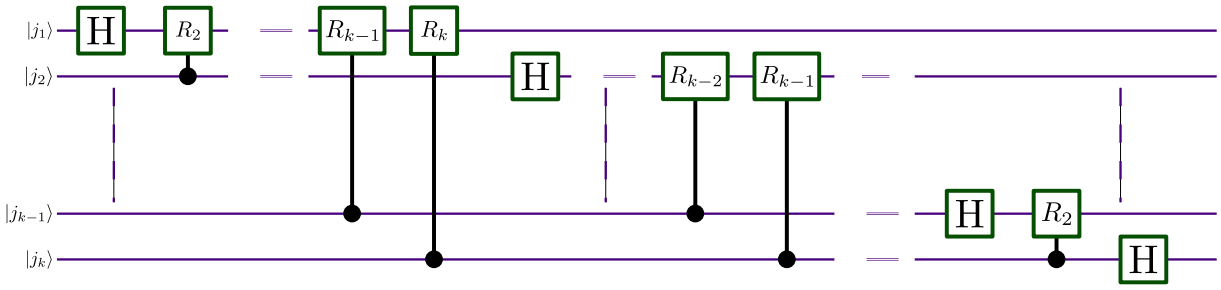

Figure 10. A circuit diagram for the quantum Fourier transform on k qubits, where R_a are rotations on qubit a, and H is the Hadamard operation. The controlled unitaries that R_a are part of are described by CR_a in equation (26).

For each desired CR_a, we perform an operation

$$CR'_a = \begin{pmatrix} e^{\pi i/2^{a+1}} & 0 & 0 & 0 \\ 0 & e^{-\pi i/2^{a+1}} & 0 & 0 \\ 0 & 0 & e^{-\pi i/2^{a+1}} & 0 \\ 0 & 0 & 0 & e^{\pi i/2^{a+1}} \end{pmatrix}. \tag{27}$$

We then correct these gates using local unitaries on each qubit, of the form

$$C_a = \begin{pmatrix} e^{-\pi i/2^{a+2}} & 0 \\ 0 & e^{3\pi i/2^{a+2}} \end{pmatrix}. \tag{28}$$

We chose to use this method of performing operations because of the limitations of the qubus quantum computer. The gates CR'_a are gates that are relatively easy to perform, and as we can apply general local unitaries the C_a gates do not present a problem.

Performing a QFT on k qubits involves a total of $\frac{1}{2}(k^2 - k)$ controlled phase gates. In a naïve case where each phase gate requires four operations to perform, we would require $2(k^2 - k)$ interactions with the bus. We can consider possible ways to reduce the total number of operations needed by using a similar technique that one described in section 3.2, since the size of the rotation we need to apply decreases exponentially with the separation of the qubits in the register. The principal differences between our technique here and our previous technique is the need to apply Hadamard gates between each set of controlled phase operations, and the local unitary corrections. As our corrections commute with the CR'_a, we do not need to perform these straightaway. However, these corrections do not commute with the Hadamard, so the corrections for each individual qubit need to be performed before the Hadamard is performed on that qubit. Multiple corrections are performed simply by taking the product of the necessary corrections for every single controlled phase gate the qubit is part of, regardless of whether it acts as a target or control qubit.

To obtain the full QFT sequence, we entangle qubit 1 with the first quadrature of the bus and then all the other $k - 1$ qubits with the second quadrature. We then disentangle qubits 1 and 2, before applying a correction and a Hadamard operation to qubit 2. Qubit 2 is then entangled to the first quadrature of the bus, thus interacting it with the other $k - 2$ qubits. We repeat this process shifting successive qubits from the second quadrature of the bus to the first, applying a correction and a Hadamard operation in between the two operations. Once qubit k has been removed from the second quadrature of the bus, it does not need to be reconnected although

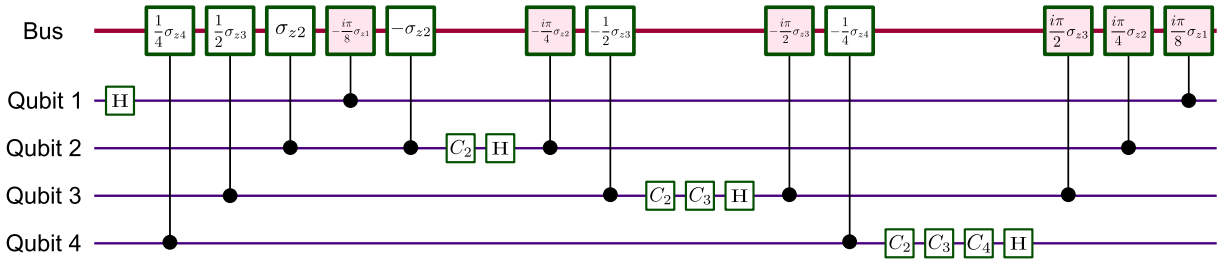

Figure 11. A diagram showing how to perform a four-qubit Fourier transform on the qubus quantum computer. The boxes show displacements performed on the continuous variable field controlled by the qubits or local operations. Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.

the correction and the Hadamard operation need to be performed. We then remove qubits 2 through $k-1$ from the first quadrature of the bus and our QFT is complete. In this sequence qubits 1 through $k-1$ interact with the first quadrature of the bus twice (once to connect them and once to remove them) and qubits 2 through k interact with the second quadrature of the bus twice. This results in a total of $4k-4$ bus operations. A diagram with suitable phase factors for performing the QFT in the four-qubit case is shown in figure 11.

We then need to consider the corrections we need to apply to our qubits. In the worst-case scenario, we would need to apply a correction for every single controlled rotation a qubit is part of, meaning that we would need k^2-k local unitaries. Since we are considering diagonal matrices, and have the ability to perform arbitrary unitaries, it is possible to reduce this down to $2k-2$ by combining the corrections performed on each qubit into those performed before the Hadamard and those performed after it. This is feasible because the matrices are diagonal, so we can work out what correction we need to apply efficiently on a classical computer. In the case of the phase estimation algorithm we are measuring in the Z-basis straight after performing our QFT. This means that it is possible to reduce the number of corrections further by simply not performing those which occur after the Hadamard. This results in a total of $k-1$ corrections being applied and therefore a total number of operations for the QFT given by

$$N_{\mathrm{FT}} = 6k - 5, \tag{29}$$

including the k necessary Hadamard gates.

7. The total number of operations needed for a simulation

We now wish to consider the total number of operations required for simulating the BCS Hamiltonian. The saving we showed in section 3 is mainly applicable to the initialization procedure; however, even for a small number of ancilla qubits the number of operations required by the phase estimation algorithm dominates to such a degree that it is only worth considering two cases: the general case and the case of limited range interactions. In the general case, we find that the total number of operations needed to implement our entire algorithm for a single run is given by

$$T_{\mathrm{G}} = P_{\mathrm{G}}(N) + SI_{\mathrm{G}}(N) + N_{\mathrm{FT}} \tag{30}$$

from equations (14), (15), (24) and (29). This gives us

$$T_{\mathrm{G}} = (2^k - 1)(6N^2 + 80N - 48) + 6k - 5 + \frac{0.1\pi}{\delta}(2N^2 + 3N + 4). \tag{31}$$

Using a similar argument for the limited range case, we find

$$T_{\mathrm{L}} = (2^k - 1)(12Np - 6p^2 - 6p + 86N - 48) + 6k - 5 + \frac{0.1\pi}{\delta}(4pN + 5N - 2p^2 - 2p + 4). \tag{32}$$

To make a comparison with Wu *et al* [19], we need to rewrite k in terms of precision. The level of precision available can be expressed as a function of 2π such that the smallest phase difference we can detect is given by $2\pi/2^k$; to have an equivalent precision to Brown *et al* [23], we want the smallest phase difference we can detect to be given by δ, therefore $2^k = 2\pi/\delta$. The total number of operations is seen to be

$$T_{\mathrm{Gk}} \approx \frac{0.1\pi}{\delta}(122N^2 + 1603N - 956) \tag{33}$$

in the general case, and

$$T_{\mathrm{Lk}} \approx \frac{0.1\pi}{\delta}(244Np - 122p^2 - 122p + 1725N - 956) \tag{34}$$

in the limited range case. We can now compare our results to those found by Wu *et al* [19], who claim to require more than $3N^4$ operations; however, this excludes the initialization procedure. Since they use the first order Trotter approximation throughout, we will derive an upper bound by using the same number of operations for the initialization procedure as the longest run of the computer. We will incorporate the precision such that

$$T_{\mathrm{NMR}} = \frac{6}{\delta}N^4 \tag{35}$$

in the nearest-neighbour case to leading order. For our qubus system in the nearest-neighbour case, we have

$$T_{\mathrm{qubus}} = \frac{\pi}{\delta}(196.9N - 120). \tag{36}$$

We can therefore see that provided $N \geqslant 5$, our qubus system requires fewer operations than an equivalent NMR simulation. For $N < 5$ it would be possible to get a saving by using a similar data extraction procedure to NMR, running the simulation for several time intervals and working out the probability of a single qubit being in $|1\rangle$. While this would require more runs and have a poor scaling, it is suitable for small systems. In the case of $N = 10$, our qubus system requires $856\,840 \approx 9 \times 10^5$ operations to simulate long-range interactions with $\delta = 0.01$, whereas the NMR system requires 6×10^6 operations to do the same for nearest-neighbour interactions. Therefore, we can already see a significant difference. Using a similar number ($\approx 6 \times 10^6$) of operations on our qubus system, it would be possible to generate operations for a BCS Hamiltonian of 97 levels in the nearest-neighbour case and 33 levels in the general case.

8. Conclusions

In this paper, we have discussed how to simulate the BCS Hamiltonian on a qubus quantum computer. Our results are also applicable to general ancilla-based schemes. We show that using

a naïve method of performing the time evolution, we require $O(N^2)$ operations, which is an $O(N^3)$ saving over a simulation on an NMR computer [19, 23]. It is possible to get further reductions in the number of gates needed in the general case, where we get almost a factor of 2 saving over our naïve method, and in certain special cases where $O(N)$ saving is possible. In the general case, we need $3N^2+5N+6$ operations, including local unitaries, for each time step of the evolution. For the special cases, which include the case where interactions are decaying exponentially with distance, only $20N-12$ operations are required for each stage in the time evolution. We show that the general case requires only six operations more than our lower bound, and that our special cases achieve their overall lower bounds. We also look at the nearest-neighbour case of the BCS Hamiltonian, which only requires $2N$ operations for U_{zz} [28] and so $11N$ operations to perform in the general case. Building upon this, we look at interactions of length p, where $p=1$ is the nearest-neighbour case, $p=2$ nearest-neighbour and next to nearest-neighbour, etc. For this case, we characterize the number of operations needed to perform each stage of the time evolution as $8N+6pN-3p^2-3p+6$.

We then apply these efficiency savings to making our operations controlled so that we can implement the phase estimation algorithm. In this case, we find that we cannot get the $O(N)$ saving over our naïve method. However, we can still obtain significant saving, and require $O(N^3)$ fewer operations than an NMR computer. As we are using the phase estimation algorithm, we also demonstrate how to obtain efficiency saving when performing a QFT on the qubus quantum computer, reducing the number of operations required by $O(N)$ to $6k-5$ per QFT. This significant improvement will allow efficient extraction of the energy gap from our system using the phase estimation algorithm. The efficiency saving for the QFT is applicable to other quantum algorithms, such as Shor's algorithm.

Wu *et al* [19] give a minimum bound on the number of operations required to simulate a BCS Hamiltonian described by 10 qubits on an NMR quantum computer. If we consider being limited to a similar number of operations, we show that on the qubus quantum computer we can simulate a BCS Hamiltonian described by 97 qubits. Thus, our qubus quantum computer has a much better efficiency than the NMR computer and hence we will be able to increase the size of the system we can simulate on an early quantum computer.

We have shown that the qubus quantum computer has significant advantages compared to an NMR computer in the number of elementary operations required; this gives $O(N^3)$ saving when simulating the BCS Hamiltonian. As we have demonstrated an $O(N)$ improvement over the naïve case of performing the QFT, we have every reason to believe that these savings will apply in a wide range of cases. Similarly, ancilla methods of quantum computing are more general than just the qubus, so these results are potentially applicable to a large range of physical systems [31]. In a previous work, we have shown similar improvements in generating cluster states [32], illustrating the generality of the techniques we have described here in the context of simulating the BCS Hamiltonian.

Acknowledgments

KLB is supported by a UK EPSRC CASE studentship from Hewlett Packard. VMK is supported by a UK Royal Society university research fellowship. WJM was supported in part by MEXT and FIRST of Japan. We thank Tim Spiller for useful discussions on the BCS Hamiltonian and the qubus architecture and Clare Horsman for the useful discussions on how to optimize the use of the qubus.

References

[1] Shor P W 1997 Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer *SIAM J. Comput.* **26** 1484–509

[2] Feynman R P 1982 Simulating physics with computers *Int. J. Theor. Phys.* **21** 467–88

[3] Lloyd S 1996 Universal quantum simulators *Science* **273** 1073–8

[4] Aspuru-Guzik A, Dutoi A D, Love P J and Head-Gordon M 2005 Simulated quantum computation of molecular energies *Science* **309** 1704–7

[5] Yu Smirnov A, Savel'ev S, Mourokh L G and Nori F 2007 Modelling chemical reactions using semiconductor quantum dots *Europhys. Lett.* **80** 67008

[6] Abrams D S and Lloyd S 1999 Quantum algorithm providing exponential speed increase for finding eigenvalues and eigenvectors *Phys. Rev. Lett.* **83** 5162

[7] Brown K L, Munro W J and Kendon V M 2010 Using quantum computers for quantum simulation *Entropy* **12** 2268–307

[8] De Raedt K, Michielsen K, De Raedt H, Trieu B, Arnold G, Richter M, Lippert Th, Watanabe H and Ito N 2007 Massive parallel quantum computer simulator *Comput. Phys. Commun.* **176** 121–36

[9] Verstraete F, Porras D and Cirac J I 2004 Density matrix renormalization group and periodic boundary conditions: a quantum information perspective *Phys. Rev. Lett.* **93** 227205

[10] Bardeen J, Cooper L N and Schrieffer J R 1957 Theory of superconductivity *Phys. Rev.* **108** 1175

[11] Richardson R W and Sherman N 1964 Exact eigenstates of the pairing-force Hamiltonian *Nucl. Phys.* **52** 221–38

[12] Anderson P W 1958 Random-phase approximation in the theory of superconductivity *Phys. Rev.* **112** 1900–16

[13] Giorgini S, Pitaevskii L P and Stringari S 2008 Theory of ultracold atomic Fermi gases *Rev. Mod. Phys.* **80** 1215–74

[14] Dean D J and Jensen M H 2003 Pairing in nuclear systems: from neutron stars to finite nuclei *Rev. Mod. Phys.* **75** 607–56

[15] Mastellone A, Falci G and Fazio R 1998 Small superconducting grain in the canonical ensemble *Phys. Rev. Lett.* **80** 4542–5

[16] Volya A, Brown A B and Zelevinsky V 2001 Exact solution of the nuclear pairing problem *Phys. Lett.* B **509** 37–42

[17] Dukelsky J and Sierra G 1999 Density matrix renormalization group study of ultrasmall superconducting grains *Phys. Rev. Lett.* **83** 172–5

[18] Ho S Y, Rowe D J and De Baerdemacker S 2010 Eigenstate estimation for the Bardeen–Cooper–Schrieffer (BCS) Hamiltonian arXiv:1011.4304

[19] Wu L-A, Byrd M S and Lidar D A 2002 Polynomial-time simulation of pairing models on a quantum computer *Phys. Rev. Lett.* **89** 057904

[20] Cleve R, Ekert A, Macchiavello C and Mosca M 1998 Quantum algorithms revisited *Proc. Roy. Soc.* A **454** 339

[21] van Loock P, Munro W J, Nemoto K, Spiller T P, Ladd T D, Braunstein S L and Milburn G J 2008 Hybrid quantum computation in quantum optics *Phys. Rev.* A **78** 022303

[22] Spiller T P, Nemoto K, Braunstein S L, Munro W J, van Loock P and Milburn G J 2006 Quantum computation by communication *New J. Phys.* **8** 30

[23] Brown K R, Clark R J and Chuang I L 2006 Limitations of quantum simulation examined by a pairing Hamiltonian using nuclear magnetic resonance *Phys. Rev. Lett.* **97** 050504

[24] Rodrigues D A, Jarvis C E A, Gyorffy B L, Spiller T P and Annett J F 2008 Entanglement of superconducting charge qubits by homodyne measurement *J. Phys.: Condens. Matter* **20** 075211

[25] Trotter H F 1959 On the product of semi-groups of operators *Proc. Am. Math. Phys.* **10** 545

[26] Suzuki M 1993 Improved Trotter-like formula *Phys. Lett.* A **180** 232–4

[27] Brown K L 2011 Using the qubus for quantum computing *PhD Thesis* University of Leeds

[28] Louis S G R, Nemoto K, Munro W J and Spiller T P 2007 The efficiencies of generating cluster states with weak nonlinearities *New J. Phys.* **9** 193

[29] Nielsen M A and Chuang I L 2000 *Quantum Computation and Quantum Information* (Cambridge: Cambridge University Press)

[30] Barenco A, Bennett C H, Cleve R, DiVincenzo D P, Margolus N, Shor P, Sleator T, Smolin J A and Weinfurter H 1995 Elementary gates for quantum computation *Phys. Rev.* A **52** 3457–67

[31] Anders J, Oi D K L, Kashefi E, Browne D E and Andersson E 2010 Ancilla-driven universal quantum computation *Phys. Rev.* A **82** 020301

[32] Horsman C, Brown K L, Munro W J and Kendon V M 2011 Reduce, reuse, recycle for robust cluster-state generation *Phys. Rev.* A **83** 042327