Genetic braid optimization: A heuristic approach to compute quasiparticle braids

Ross B. McDonald[1] and Helmut G. Katzgraber[1,2]

[1] *Department of Physics and Astronomy, Texas A&M University, College Station, Texas 77843-4242, USA*
[2] *Theoretische Physik, ETH Zurich, CH-8093 Zurich, Switzerland*

(Dated: February 14, 2013)

In topologically protected quantum computation, quantum gates can be carried out by adiabatically braiding two-dimensional quasiparticles, reminiscent of entangled world lines. Bonesteel *et al.* [Phys. Rev. Lett. **95**, 140503 (2005)], as well as Leijnse and Flensberg [Phys. Rev. B **86**, 104511 (2012)], recently provided schemes for computing quantum gates from quasiparticle braids. Mathematically, the problem of executing a gate becomes that of finding a product of the generators (matrices) in that set that approximates the gate best, up to an error. To date, efficient methods to compute these gates only strive to optimize for accuracy. We explore the possibility of using a generic approach applicable to a variety of braiding problems based on evolutionary (genetic) algorithms. The method efficiently finds optimal braids while allowing the user to optimize for the relative utilities of accuracy and/or length. Furthermore, when optimizing for error only, the method can quickly produce efficient braids.

PACS numbers: 03.67.Lx, 75.40.Mg, 73.43.-f

I. INTRODUCTION

Sensitivity to noise makes most of the current quantum computing schemes prone to error and nonscalable, allowing only for small proof-of-principle devices. Topological quantum computation[1,2] offers an elegant alternative to overcome decoherence by using non-Abelian quasiparticles. Materials with sufficiently complex topologically ordered phases can thus be used as media for intrinsically fault-tolerant and scalable quantum information processing. Different proposals and implementations exist to date, ranging from fractional quantum Hall systems[3–5] with filling factors $\nu = 5/2$ and $\nu = 12/5$, conjectured to exhibit non-Abelian anyonic excitations,[6,7] to quantum dimer models[8–14] (implemented via Josephson junction arrays),[10,15] spin and Hubbard models,[16–19] toric[2] and color[20] codes, and an anisotropic spin model,[21] as well as implementations using cold atomic[22] or molecular[23,24] gases. While the holy grail is the firm establishment of a phase with non-Abelian anyons, virtually all aspects of topological quantum computation are now under intense experimental and theoretical study. Unfortunately, huge technological challenges remain for the development of working devices; however, some proposals based on current technologies exist.[10,15]

While bosons or fermions pick up phase factors of ± 1 when braided, for anyonic particles these simple phases are replaced by non-Abelian matrices. The matrices act on a (degenerate) Hilbert space with a dimensionality that grows exponentially in the number of quasiparticles and whose states are intrinsically immune to decoherence because they cannot be distinguished by local measurements. A topologically protected quantum gate then can be implemented by adiabatically braiding quasiparticles.[1,2,25,26] At low enough temperatures the system is, by design, protected from decoherence because errors only can occur due to particle-hole excitations.

There are different quasiparticle systems that can be used to generate quantum gates. However, in all cases the problem can be reduced to finding a product of braid generators (matrices) that approximates a given quantum gate with the smallest possible error and, if possible, as short as possible. For example, in Bonesteel *et al.*[25] braids of Fibonacci anyons[1] are computed. The fusion rules for these anyons make the Hilbert space of the quasiparticles two-dimensional (see Sec. II for details); i.e., a product of two-dimensional matrices has to be computed. Bonesteel *et al.* first performed a brute force (exhaustive) search up to a braid length of 46 exchanges, obtaining nontrivial gates with an error $\varepsilon \sim 10^{-3}$. Unfortunately, the search space grows exponentially with the length of the braid. Using the Solovay-Kitaev algorithm,[27–29] they then were able to compute braids to any desired accuracy with a length that grows $\sim |\log_{10}(\varepsilon)^4|$. For example, for an accuracy of $\varepsilon \sim 10^{-3}$ the Solovay-Kitaev algorithm would require braids of an approximate length of 81 exchanges. However, the Solovay-Kitaev algorithm does not allow for the user to optimize for the relative utilities of accuracy vs length. Depending on physical implementations, a longer braid might be more problematic due to error proliferation, and as such, having the option to either optimize for accuracy and/or length might lead to braids better suited for a given physical implementation.

In this paper we explore the possibility of using evolutionary (genetic) algorithms[30] to efficiently find optimal braids while allowing the user to optimize for the relative utilities of accuracy and/or length. We test the method with the braids computed by Bonesteel *et al.*,[25] as well as a recent proposal by Leijnse and Flensberg[31] that braids six Majorana fermions to create two-qubit gates. Furthermore, we show that when optimizing for error only, the method can quickly produce efficient braids, outperforming brute force searches. We emphasize that the presented method is generic and therefore can be applied to any problem that requires the computation of the optimal product of (non-Abelian) operators. Thus it can

be applied, for example, also to surface codes.[32]

This paper is structured as follows. In Sec. II we show how the complex quantum computing problem can be reduced to a simple mathematical problem of finding matrix products, followed by a brief review of previous methods in Sec. III. The evolutionary algorithm is introduced in Sec. IV, followed by results in Sec. V and concluding remarks.

II. SIMPLIFIED PROBLEM REPRESENTATION

In this section we illustrate the method on two different quantum computing proposals.

A braid operation can be represented by a matrix that acts on the qubit space. These matrices will be referred to as *generators*, and the quantum gate that a braid represents is the product of the generators that represent the individual braid operations. The problem of finding braiding operations that approximate gates is then reduced to finding a product chain of the reduced generators and their inverses that approximates the matrix representing the quantum gate.

Fibonacci anyon braids[25] only encompasses one-qubit gates. In such systems, the braid transition operators result in a phase change for the noncomputational state, and therefore it can be ignored. Overall, phases in the problem can also be ignored. Therefore the transition matrices can be projected onto SU(2) by a multiplication with $e^{\imath\pi/10}$, yielding for the generators and their graphical representations

$$\sigma_1 = \begin{pmatrix} e^{-\imath 7\pi/10} & 0 \\ 0 & -e^{-\imath 3\pi/10} \end{pmatrix} =$$

$$\sigma_2 = \begin{pmatrix} -\tau e^{-\imath\pi/10} & -\imath\sqrt{\tau} \\ -\imath\sqrt{\tau} & -\tau e^{\imath\pi/10} \end{pmatrix} =$$

where $\tau = (\sqrt{5}-1)/2$, and the graphical representations are those used, for example, in Fig. 3.

In the Leijnse and Flensberg scheme based on Majorana fermions the braid operators act on a two-qubit system; i.e., the gates will be two-qubit gates. The generators for this scheme are higher-dimensional, i.e.,

$$B_1 = \begin{pmatrix} \imath & 0 & 0 & 0 \\ 0 & \imath & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \; B_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & \imath & 0 \\ 0 & 1 & 0 & \imath \\ \imath & 0 & 1 & 0 \\ 0 & \imath & 0 & 1 \end{pmatrix}, \; B_3 = \begin{pmatrix} \imath & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \imath \end{pmatrix}, \; B_4 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & \imath & 0 & 0 \\ \imath & 1 & 0 & 0 \\ 0 & 0 & 1 & -\imath \\ 0 & 0 & -\imath & 1 \end{pmatrix}, \; B_5 = \begin{pmatrix} \imath & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \imath & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (1)$$

The goal is now to find a product of generator matrices that produces a braid that represents a gate operation under the constraints that either length is minimized, error is minimized, or both length and error are minimized.

III. TRADITIONAL APPROACHES

The naïve approach to solve the braiding problem is a brute-force search. A target error is set, and the set of all braids is searched from shortest to longest until a braid whose error is smaller than or equal to the target error is found. However, this approach is nonscalable, as illustrated in Fig. 1 for Fibonacci anyons. In this case we have four possible matrices (two generators and their inverses) for each position in the braid (two or three choices if cancellations between inverses are not ignored). This means that the number of different braids of length ℓ is 4^ℓ (or in the range 2^ℓ – 3^ℓ including cancellations). This is even worse in the Leijnse and Flensberg scheme for Majorana fermions where one has 10 different matrices; i.e., an exhaustive search for a braid of length ℓ might have a worst-case run time of order $O(10^\ell)$. Because the number of possible braids grows exponentially with length, a brute-force search would be too slow for most practical applications. Note, however, that a bidirectional search[33–35] greatly improves the performance.

The Solovay-Kitaev algorithm provides a boost in the efficiency of finding more accurate braids, but at the cost of accepting braids that are longer than necessary. Depending on the implementation, this might be problematic: While a given error might be desirable, a given hardware implementation might degrade considerably with the length of the braid. In such cases short braids might be desirable. Given a target error of ε, the Solovay-Kitaev algorithm produces braids of length $O[\log_{10}^{3.97}(1/\varepsilon)]$ that are guaranteed to have an error less than ε in a run time of $O[\log_{10}^{2.71}(1/\varepsilon)]$.[28,29]

Another option explored by Burrello *et al.*[36] is braid hashing, in which approximations of the identity braid are used to refine crude approximations of the target braid into more accurate solutions. This method is fast and can produce very accurate braids, but it does not address the problem of increasing braid length with accuracy. Furthermore, the braid hashing algorithm works only for Fibonacci anyons, and it is unclear how the method can be generalized to other systems, such as Majorana fermions.

In particular, none of these methods seek to optimize braid length in addition to accuracy except for brute

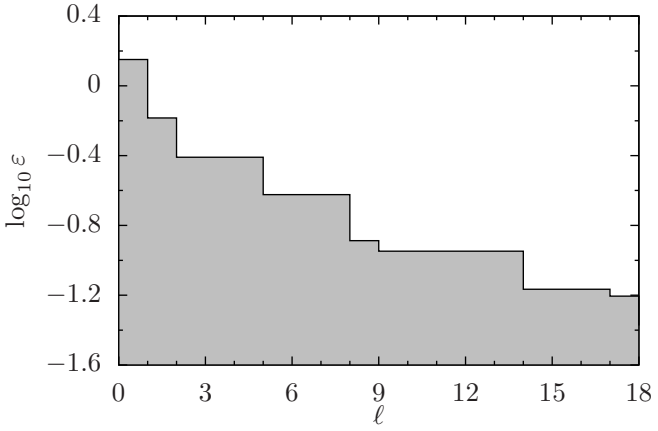

FIG. 1: Minimum possible error ε for a braid of maximum length ℓ of Fibonacci anyons. This brute-force result for braids up to $\ell = 18$ is a lower bound for the error for all other search methods. The line corresponds to the optimum; i.e., any heuristic method can only produce a solution that lies on or above the line.

force; however, brute force is slow. Below we present an efficient algorithm that can be tuned to optimize for both length and/or accuracy. The method has the potential to create shorter braids than the Solovay-Kitaev algorithm. However, we note that the Solovay-Kitaev algorithm is faster and can overcome some convergence problems the genetic approach faces (see below). In comparison to brute-force methods, the genetic approach yields good results considerably faster. We also emphasize that this method is applicable to any system where quantum gates are built from a finite set of fault-tolerant gates. Although there have been some attempts to solve this problem generically, the approach of, e.g., Ref. 32 only applies to single-qubit systems. The genetic method outlined below can be potentially applied to arbitrary systems.

IV. EVOLUTIONARY ALGORITHM

Mathematically, the problem at hand is similar to solving a Rubik's cube: The goal is to find the shortest set of matrix operations (cube rotations) to obtain the minimum of a cost function (uniformly colored faces of the cube), which here represents the shortest braid or the smallest error.

The proposed algorithm resembles a steady-state genetic algorithm: A population of random-solution braids is generated in the initialization step of the algorithm. The population then evolves in an iterative process where different generations are developed according to predefined mathematical operations on the population. Each update on a generation is broken into two steps, *culling* and *breeding*, which are described in detail below. After a predefined number of generations have been executed, the algorithm terminates, and the best braid encountered by the algorithm is presented as the solution. A "best braid" is defined as the braid with the highest fitness, i.e., the braid that minimizes the problem-dependent cost function (described below) for the problem.

To simplify matters, we introduce the following notation: Let $B^{[a,b]}$ denote the sub braid of braid B from the ath element to bth element (inclusive), and let $B^{[a,]}$ denote the sub braid of B from the ath element to the end of B. Furthermore, let the concatenation of braid variables indicate a concatenation of the actual braids. For example, $B = B_1B_2$ means that B is a concatenation of braids B_1 and B_2. $B = B_1^{[3,]}B_2^{[2,5]}$ would indicate that B is a concatenation of the third to end sub braids of B_1 and the second to fifth sub braids of B_2. Let len() be a function that has a braid as its argument and returns that braid's length, let mat() be a function that has a braid as its argument and returns the product of the braid's elements in matrix form, and let d() be a function that evaluates the distance between two braids, specifically,

$$\mathrm{d}(B_1, B_2) = |\mathrm{mat}(B_1) - \mathrm{mat}(B_2)|\,. \tag{2}$$

Here and for the rest of the paper, the matrix norm used is

$$|X| = \sqrt{\sum_{ij} X_{ij}^2}. \tag{3}$$

Equation (2) defines the metric used to determine the "distance" between two braids.

A high-level view of the simulation is provided by the following pseudocode in which m is the population size, *generations* is the number of generations to evolve the population, and *best* is the current best braid:

population ← new population of size m
for $i = 1 \rightarrow$ *generations* **do**
  Sort *population* ascending by fitness
  if $fitness(best) < fitness(population[m])$ **then**
    $best \leftarrow population[m]$
  end if
  Perform culling (least fit 10% removed)
  Repopulate missing 10% with breeding
  $i \leftarrow i + 1$
end for

It should be noted that in this pseudocode and all following pseudocodes, collections use one-based indices (i.e., *population*[1] is the first element of population). At the end of a generation the population is the same size as in the beginning; however, we expect that the offsprings are fitter than the initial randomly chosen parents.

A. Culling

The population is sorted according to a fitness function. The fitness can, in general, be any real-valued function of braid length ℓ and error ε. Here we use

$$f(\ell, \varepsilon) = \frac{1-\lambda}{1+\varepsilon} + \frac{\lambda}{\ell}. \tag{4}$$

The braid error is calculated with the following metric:

$$\varepsilon = |B - X|, \tag{5}$$

where B is a braid matrix and X represents the target matrix (gate to be emulated). The parameter λ allows one to tune between a short braid or a more accurate braid; i.e., for $\lambda \to 1$ the system is tuned for length, whereas for $\lambda \to 0$ the system is tuned for error reduction.

After sorting the population by fitness, the bottom 10% of the genes (braids) are removed.

B. Breeding

After a culling operation only 90% of the genes are left in the gene pool. The remaining 10% are filled by combining the remaining braids into new braids, i.e., breeding. From the 90% of the braids that survived the culling operation, which represents the top 90% of the genes in the population, two braids are selected as parents of a new braid for the gene pool. (Note that values ∼90% are typically used in the literature). Let these parents be denoted as P_1 and P_2, and let the offspring be denoted as C_1 and C_2. The way two parent braids are combined into an offspring plays a *crucial* role in the efficiency of the algorithm.

Our initial rather naïve approach was to select two random integers n_1 and n_2 such that $n_1 \in (1, \text{len}(P_1)]$ and $n_2 \in (1, \text{len}(P_2)]$. The boundaries of these ranges are chosen to prevent duplication of the parents. The offspring are then formed as $C_1 = P_1^{[1,n_1-1]} P_2^{[n_2,]}$ and $C_2 = P_2^{[1,n_2-1]} P_1^{[n_1,]}$. Due to the noncommutativity of matrix multiplication, this method is no better than *randomly generating* two offspring. To remedy this problem, we used a different recombination method, referred to as *contextual recombination*. In contextual recombination the partition points n_1 and n_2 are chosen by minimizing the distance between the first halves of the two parent braids. However, to avoid cloning, we require that the first halves not be identical. The actual recombination method after n_1 and n_2 are chosen is the same as above. To choose n_1 and n_2, one must first determine m, where m is the largest integer such that $P_1^{[1,m]}$ and $P_2^{[1,m]}$ both exist and are identical. Once m is determined, n_1 and n_2 are chosen such that $n_1 \in (m, \text{len}(P1)]$, $n_2 \in (m, \text{len}(P_2)]$, and $\text{d}(P_1^{[1,n_1-1]}, P_2^{[1,n_2-1]})$ is minimized. These values can be determined using the following pseudocode:

```
minDistance ← ∞
m ← 0
while m < min(len(P1), len(P2)) and P1^[m,m] = P2^[m,m] do
    m ← m + 1
end while
for i = m → len(P1) do
    for j = m → len(P2) do
        dist ← d(P1^[1,i−1], P2^[1,j−1])
        if dist < minDistance then
            minDistance ← dist
            n1 ← i
            n2 ← j
        end if
    end for
end for
```

V. RESULTS

We have run the algorithm for both Fibonacci anyons and Majorana particles. The population sizes are 80 individuals for both cases following the recommendations by Belmont-Moreno.[37] Increasing the population size showed no significant improvement on the results but increased the run time.

We start with Fibonacci anyons as studied in Ref. 25 with $\lambda = 0$. In this case we only optimize for accuracy and not for length. The goal is to emulate the X-rotation gate

$$X = \begin{pmatrix} 0 & \imath \\ \imath & 0 \end{pmatrix}. \tag{6}$$

In the two-qubit case we emulate the controlled NOT (or CNOT) gate, namely

$$X = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{7}$$

Figure 2 shows the error ε as a function of the run time of the algorithm. The method is capable of improving solutions very quickly (after approximately 150 generations) but is unable to improve after a certain point. An interesting example braid of the algorithm that shows its potential can be seen in Fig. 3. This is very promising, and we believe the minimum error problem can be solved by introducing a clever mutation method. However, our attempts to implement a basic mutation (i.e., changing a generator into a random new one and inserting approximations to the identity braid into the braid) rendered the algorithm as inefficient as a random search.

Unfortunately, for two-qubit gates (Majorana fermions) the algorithm is not efficient. Figure 2 shows the error ε as a function of the run time (dashed line). The data converge quickly to a plateau and cease to improve; i.e., the accuracy of the braid cannot be improved.

Figure 4 shows that by tuning λ in the fitness function we are able to tune fitness against accuracy effectively for the case of Fibonacci anyons emulating the X-rotation gate. The squares represent averages over 100 runs and the ellipses correspond to standard deviations. The variance for small λ is large. However, for larger values of λ the length of the braid can be effectively constrained.

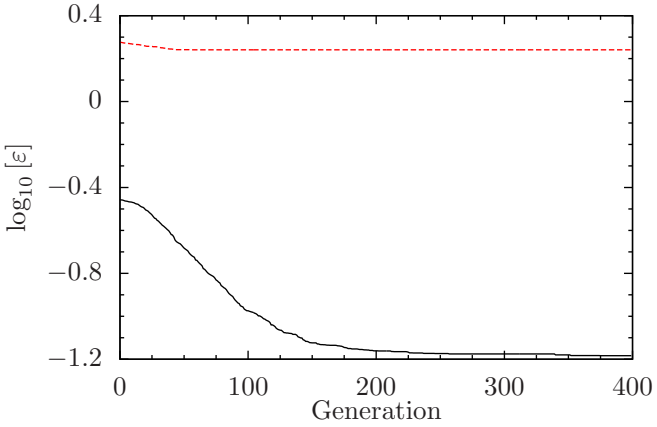

FIG. 2: (Color online) Average error in the population for each generation averaged over 100 sample runs of the algorithm for $\lambda = 0$. The solid line represents the average error for Fibonacci anyons when emulating an X-rotation gate. The dashed line is for Majorana particles emulating a CNOT gate.

$$= \sigma_2^{-2}\sigma_1^4\sigma_2^{-1}\sigma_1\sigma_2^{-1}\sigma_1\sigma_2\sigma_1^{-2}\sigma_2\sigma_1^{-1}\sigma_2^{-5}\sigma_1\sigma_2^{-1} \approx \begin{pmatrix} 0 & i \\ i & 0 \end{pmatrix}$$

FIG. 3: (Color online) Example result for a braid of Fibonacci anyons emulating the X-rotation gate. The error of this approximation is 3.1×10^{-3}.

Although the spread in the accuracy is large, repeating the simulation multiple times allows one to determine an optimal braid with a small error and small length quickly (less than 1h for 100 runs on an average CPU). We expect that by introducing clever mutations the spread in the data can be reduced.

Figure 4 also shows that for high values of λ, the algorithm produces results very near the best-case boundary but is constrained to the high-error region. As λ decreases, the solutions move away from the best-case boundary, producing longer-than-needed braids.

VI. CONCLUSIONS

We have introduced a *generic* algorithm based on evolutionary methods to approximate gates using quasiparticle braids. While single-qubit braids of Fibonacci anyons can be computed efficiently, the method fails to produce optimal braids for two-qubit gates. The latter presents an unresolved challenge that we will attempt to tackle in the near future. Our results suggest that mutations might be key in the improvement of the method.

We emphasize that the developed method is generic and therefore can be applied to problems ranging from general quantum compiling, to orienting devices using coarse stepper motors in industrial applications, as well as generic optimization of problems with competing goals. It would be interesting to compare our results to bidirectional search,[33–35] which we plan to do in the future.

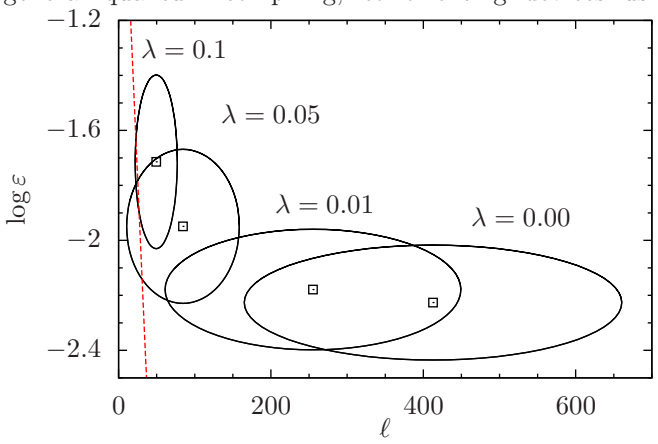

FIG. 4: (Color online) Distribution of the output of the algorithm for different values of λ. Each ellipse represents the output distribution for a single value of λ centered on the average output, with the ellipse bounds being a single standard deviation from the mean. Averages are over 100 runs. The dashed red line is an extrapolation of the graph in Fig. 1.

Acknowledgments

We would like to thank N. Bonesteel for useful discussions. H.G.K. acknowledges support from the Swiss National Science Foundation (Grant No. PP002-114713) and the National Science Foundation (Grant No. DMR-1151387).

[1] M. H. Freedman, A. Y. Kitaev, M. J. Larsen, and Z. Wang, *Topological Quantum Computation* (2002), (arXiv:quant-ph/0101025).
[2] A. Y. Kitaev, *Fault-tolerant quantum computation by anyons*, Ann. Phys. **303**, 2 (2003).
[3] G. Moore and N. Read, *Nonabelions in the fractional quantum hall effect*, Nucl. Phys. B **360**, 362 (1991).
[4] D. Arovas, J. R. Schrieffer, and F. Wilczek, *Fractional Statistics and the Quantum Hall Effect*, Phys. Rev. Lett. **53**, 722 (1984).

[5] F. E. Camino, W. Zhou, and V. J. Goldman, *Aharonov-Bohm Superperiod in a Laughlin Quasiparticle Interferometer*, Phys. Rev. Lett. **95**, 246802 (2005).

[6] P. Bonderson, A. Kitaev, and K. Shtengel, *Detecting Non-Abelian Statistics in the $\nu = 5/2$ Fractional Quantum Hall State*, Phys. Rev. Lett. **96**, 016803 (2006).

[7] F. E. Camino, W. Zhou, and V. J. Goldman, *Transport in the Laughlin quasiparticle interferometer: Evidence for topological protection in an anyonic qubit*, Phys. Rev. B **74**, 115301 (2006).

[8] D. S. Rokhsar and S. A. Kivelson, *Superconductivity and the quantum hard-core dimer gas*, Phys. Rev. Lett. **61**, 2376 (1988).

[9] R. Moessner and S. L. Sondhi, *Resonating Valence Bond Phase in the Triangular Lattice Quantum Dimer Model*, Phys. Rev. Lett. **86**, 1881 (2001).

[10] L. B. Ioffe, M. V. Feigel'man, A. Ioselevich, D. Ivanov, M. Troyer, and G. Blatter, *Topologically protected quantum bits from Josephson junction arrays*, Nature **415**, 503 (2002).

[11] G. Misguich, D. Serban, and V. Pasquier, *Quantum Dimer Model on the Kagome Lattice: Solvable Dimer-Liquid and Ising Gauge Theory in strongly correlated systems*, Phys. Rev. Lett. **89**, 137202 (2002).

[12] P. Fendley and E. Fradkin, *Realizing non-Abelian statistics*, Phys. Rev. B **72**, 024412 (2005).

[13] A. Ralko, M. Ferrero, F. Becca, D. Ivanov, and F. Mila, *Zero-temperature properties of the quantum dimer model on the triangular lattice*, Phys. Rev. B **71**, 224109 (2005).

[14] R. Moessner and S. L. Sondhi, *Three-dimensional resonating-valence-bond liquids and their excitations*, Phys. Rev. B **68**, 184512 (2003).

[15] A. F. Albuquerque, H. G. Katzgraber, M. Troyer, and G. Blatter, *Engineering Exotic Phase for Topologically-Protected Quantum Computation by Emulating Quantum Dimer Models*, Phys. Rev. B **78**, 014503 (2008).

[16] L. Balents, M. P. A. Fisher, and S. M. Girvin, *Fractionalization in an easy-axis Kagome antiferromagnet*, Phys. Rev. B **65**, 224412 (2002).

[17] M. H. Freedman, C. Nayak, and K. Shtengel, *Extended Hubbard Model with Ring Exchange: A Route to a Non-Abelian Topological Phase*, Phys. Rev. Lett. **94**, 066401 (2005).

[18] D. N. Sheng and L. Balents, *Numerical Evidences of Fractionalization in an Easy-Axis Two-Spin Heisenberg Antiferromagnet*, Phys. Rev. Lett. **94**, 146805 (2005).

[19] S. V. Isakov, Y. B. Kim, and A. Paramekanti, *Spin-Liquid Phase in a Spin-1/2 Quantum Magnet on the Kagome Lattice*, Phys. Rev. Lett. **97**, 207204 (2006).

[20] H. Bombin and M. A. Martin-Delgado, *Topological Quantum Distillation*, Phys. Rev. Lett. **97**, 180501 (2006).

[21] A. Kitaev, *Anyons in an exactly solved model and beyond*, Ann. Phy. **321**, 2 (2006).

[22] H. P. Büchler, M. Hermele, S. D. Huber, M. P. A. Fisher, and P. Zoller, *Atomic quantum simulator for lattice gauge theories and ring exchange models*, Phys. Rev. Lett. **95**, 040402 (2005).

[23] A. Micheli, G. K. Brennen, and P. Zoller, *A toolbox for lattice-spin models with polar molecules*, Nat. Phys. **2**, 341 (2006).

[24] G. Pupillo, A. Micheli, H. P. Büchler, and P. Zoller, *Condensed Matter Physics with Cold Polar Molecules* (2008), (arXiv:cond-mat/0805.1896).

[25] N. E. Bonesteel, L. Hormozi, G. Zikos, and S. H. Simon, *Braid Topologies for Quantum Computation*, Phys. Rev. Lett. **95**, 140503 (2005).

[26] H. Xu and J. M. Taylor, *Unified approach to topological quantum computation with anyons: From qubit encoding to Toffoli gate*, Phys. Rev. A **84**, 012332 (2011).

[27] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000).

[28] C. M. Dawson and M. A. Nielsen, *The Solovay-Kitaev algorithm*, Quantum Information & Computation **6**, 81 (2006).

[29] L. Hormozi, G. Zikos, N. E. Bonesteel, and S. H. Simon, *Topological Quantum Compiling*, Phys. Rev. B **75**, 165310 (2007).

[30] A. K. Hartmann and H. Rieger, *New Optimization Algorithms in Physics* (Wiley-VCH, Berlin, 2004).

[31] M. Leijnse and K. Flensberg, *Using hybrid topological-spin qubit systems for two-qubit-spin gates*, Phys. Rev. B **86**, 104511 (2012).

[32] A. Bocharov and K. M. Svore, *Resource-Optimal Single-Qubit Quantum Circuits*, Phys. Rev. Lett. **109**, 190501 (2012).

[33] H. Xu and X. Wan, *Constructing functional braids for low-leakage topological quantum computing*, Phys. Rev. A **78**, 042325 (2008).

[34] L. Hormozi, N. E. Bonesteel, and S. H. Simon, *Topological Quantum Computing with Read-Rezayi States*, Phys. Rev. Lett. **103**, 160501 (2009).

[35] G. Zikos, K. Yang, N. E. Bonesteel, L. Hormozi, and S. H. Simon, *Braiding and Entanglement in Nonabelian Quantum Hall States*, Int. J. Mod. Phys. B **23**, 2727 (2009).

[36] M. Burrello, H. Xu, G. Mussardo, and X. Wan, *Quantum Hashing with the Icosahedral Group*, Phys. Rev. Lett. **104**, 160502 (2010).

[37] E. Belmont-Moreno, *The Role of Mutation and Population Size in Genetic Algorithms Applied to Physics Problems*, Int. J. of Mod. Phy. C. **12**, 1345 (2001).