Universal quantum computation by discontinuous quantum walk

Michael S. Underwood* and David L. Feder†
Institute for Quantum Information Science, University of Calgary, Calgary, Alberta T2N 1N4, Canada
(Received 18 June 2010; published 7 October 2010)

Quantum walks are the quantum-mechanical analog of random walks, in which a quantum "walker" evolves between initial and final states by traversing the edges of a graph, either in discrete steps from node to node or via continuous evolution under the Hamiltonian furnished by the adjacency matrix of the graph. We present a hybrid scheme for universal quantum computation in which a quantum walker takes discrete steps of continuous evolution. This "discontinuous" quantum walk employs perfect quantum-state transfer between two nodes of specific subgraphs chosen to implement a universal gate set, thereby ensuring unitary evolution without requiring the introduction of an ancillary coin space. The run time is linear in the number of simulated qubits and gates. The scheme allows multiple runs of the algorithm to be executed almost simultaneously by starting walkers one time step apart.

DOI: 10.1103/PhysRevA.82.042304 PACS number(s): 03.67.Ac, 05.40.Fb

I. INTRODUCTION

In analogy to the use of random walks to speed up classical computation [1], the role of quantum walks has been explored in the realm of quantum computation [2,3]. Quantum walks were first applied to quantum algorithms known to be more efficient than their classical counterparts, such as Grover's search of an unsorted array [4,5], the element distinctness problem [6], and triangle finding [7] and its extension to k-cliques [8]. It was quickly recognized that quantum walks could also be used to generate quantum algorithms for various problems more directly than was possible within the context of the conventional quantum-circuit model, for example traversing glued binary trees [9,10] and evaluating decision trees [11], NAND trees [12,13], and game trees (AND-OR formulas) [14].

More recently, quantum walks have been shown to be computationally universal in both the continuous-time [15] and discrete-time [16] formulations. In both cases, the walker moves from left to right along "rails" or lines of vertices, labeled by computational basis states. These rails are interspersed with small graphs, or "widgets," that transform the state of the quantum walker in analogy to gates in the circuit model. The widgets are attached either to individual rails or between pairs of rails, and the transformations are chosen in such a way as to effect a desired computation. The collection of rails and widgets forms a computational graph, which mimics the circuit model via the unitary evolution of the walker in its spatial Hilbert space.

The continuous-time model for universal computation proposed in Ref. [15] makes use of a walker with a tightly peaked momentum profile. This requires each of its rails to include semi-infinite "tails" (linear graphs) both before and after the computational graph, though in practice the length of these tails needs only to be large compared to twice the total evolution time of the walker within the graph (i.e., proportional to the circuit depth). Additionally, the preparation of the momentum state requires an initial sequence of momentum filter widgets, each with its own tail. A side effect is that most of the walker's probability never enters the computational graph. While these considerations only increase the resources polynomially in the number of widgets used, their presence makes the scheme somewhat cumbersome.

In the discrete-time scheme of Ref. [16], double-edged rails are employed in order to guarantee that the walker moves through the entire graph strictly from left to right. Each vertex is attached to four edges, two of which are connected to the vertex to the left, and two to the right. While this scheme does not require tails, the walker must have at least eight internal states because the various widgets require two-, four-, and eight-dimensional coins.

An alternative approach to universal quantum computation discussed in this work is based on perfect state transfer (PST) [17,18]. In PST, quantum states are transferred perfectly between two nodes of a graph in continuous time. While PST was originally described in terms of spin chains [19], it has more recently been extended to continuous-time quantum walks on graphs [20–22]. In the latter formulation of PST, the walker's state at the output vertex is identical to that at the input, modulo a phase. In order to simulate a universal quantum circuit for n qubits, one would need to construct a graph in which an input state on 2^n vertices could be transferred via PST to 2^n vertices, together with the desired 2^n-dimensional unitary operator U. While this task appears to be difficult for general U, it might be possible to decompose the graph into a small set of widgets, each of which individually allows for PST or a straightforward extension of it. Alternating these widgets with some other set of processes could then result in universal quantum computation [23] in a manner analogous to the interleaving of driving and query Hamiltonians that can efficiently simulate continuous-time quantum query algorithms under the discrete query model [24].

We present a hybrid scheme for universal quantum computation that combines the best features of the continuous- and discrete-time schemes discussed above while minimizing their disadvantages: the walker undergoes PST under continuous evolution, but only in discrete steps. In this "discontinuous quantum walk," graphs are turned on and off at discrete time intervals in a prescribed manner. The walker moves through these graphs in sequence, resulting in the implementation

*munderwood@qis.ucalgary.ca
†Corresponding author: dfeder@ucalgary.ca

of the desired 2^n-dimensional unitary U. In the absence of errors, the initial state propagates through the graph without loss of amplitude at the output; furthermore, no coin degree of freedom is required even though the procedure utilizes discrete time steps. The scheme has the added advantage that new walkers can be sent through the same graph at regular intervals, allowing for nearly simultaneous repetition of the algorithm with no additional overhead.

The remainder of this paper is organized as follows. In Sec. II we define the hybrid scheme for universal quantum computation via discontinuous quantum walk. In Sec. III we describe a set of fundamental elements that fulfill the requirements of the scheme, and in Sec. IV we show how to combine them to create a universal set of gates. In Sec. V we provide some concluding remarks.

II. HYBRIDIZING DISCRETE- AND CONTINUOUS-TIME WALKS

A quantum walk takes place on a graph $\mathcal{G} = (V, E)$, where V is a set of vertices and $E \subseteq V \times V \times W$ is a set of edges defined by pairs of elements of V and associated edge weights w_{ij} taken from $W = \{w_{ij}\}$. Often W is simply the single-element set $\{1\}$, in which case it need not be present, but more generally it can be any set of numbers. An undirected weighted graph $\mathcal{G}$ is defined by a corresponding adjacency matrix G, with matrix elements defined by

$$G_{ij} = \begin{cases} w_{ij}, & (i,j,w_{ij}) \in E, \\ 0 & \text{otherwise}, \end{cases} \tag{1}$$

where $w_{ij} = w_{ji} > 0$. By definition, G is real and symmetric and can therefore be interpreted as a Hamiltonian on the state space $\mathcal{V} = \{|v\rangle : v \in V\}$. Doing so describes a continuous-time quantum walk on $\mathcal{G}$, where a quantum walker initially on vertex v_I in the state $|I\rangle = |v_I\rangle$ evolves in time t to the final state $|F\rangle = \exp(-\mathrm{i}Gt)|v_I\rangle$, which is generally a superposition of vertex states $|v\rangle \in \mathcal{V}$.

With perfect quantum state transfer (PST), the final state after a time $t_0 > 0$ corresponds to unit probability on a single vertex v_F, $|F\rangle = |v_F\rangle$. In particular, a line of M segments exhibits PST from one end to the other for the particular choice of edge weights $w_{i,i+1} = \sqrt{i(M+1-i)}$ for $i \in \{1,\ldots,M\}$ [19]; an example is shown in Fig. 1(a). In this situation, a

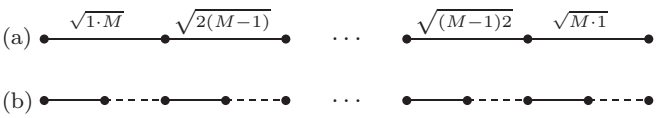

FIG. 1. Two different methods to effect perfect state transfer. (a) A line of M segments weighted so as to provide perfect state transfer from one end to the other in time $\pi/2$. All line segments are always "on" but have varying weights $w_{i,i+1} = \sqrt{i(M+1-i)}$, $i \in \{1,\ldots,M\}$. (b) A line of M segments with unit weight, with the solid and dashed couplings turned on alternately so as to provide perfect state transfer from one end to the other in time $M\pi/2$; that is, the coupling between a pair of adjacent nodes is alternated between 0 and 1.

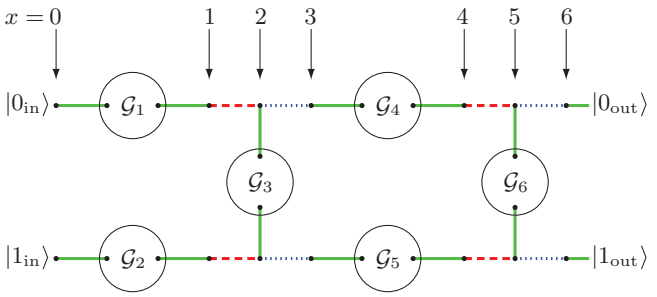

FIG. 2. (Color online) Implementation of single-qubit operations. The solid (green), dashed (red), and dotted (blue) lines form three distinct disconnected graphs, $\mathcal{G}_g$, $\mathcal{G}_r$, and $\mathcal{G}_b$, respectively; these graphs are switched on and off via global control in an algorithm-independent sequence, namely: g, r, g, b, repeat. The graphs $\mathcal{G}_i$, $i \in \mathbb{Z}$, are determined by the desired algorithm and are chosen from a specified set. The result is that a walker initially at $x = 0$ will take discrete steps to successive values of x, being transformed in the process.

walker initially localized at time $t = 0$ on the leftmost node, $|I\rangle = |v_0\rangle$, will be localized on the rightmost node at time $t = \pi/2$; a phase of $(-\mathrm{i})^M$ will have been applied to the state. Consider instead the combination of line segments in Fig. 1(b). If the walker begins on the leftmost node and the dashed line segments are disabled (i.e., their weight or coupling constant vanishes), then after a time $t = \pi/2$ the walker will have transferred perfectly to the second node and acquired a phase of $-\mathrm{i}$. If at this point the solid lines are switched off and the dashed ones enabled, the walker will proceed to the third node. In this manner it can be perfectly transferred to the rightmost node in M discrete steps, taking a total time of $M\pi/2$. Note that the scheme requires only two different unconnected graphs: those with solid and dashed edges shown in Fig. 1(b), which are enabled in an alternating pattern. The direction the walker travels on these "transport" rail segments then depends crucially on the initial occupied node.

The representation of a qubit requires two such rails, with one encoding the logical $|0\rangle$ state and the other the logical $|1\rangle$. Operations on the qubit are effected by interspersing the transport segments with widgets that transform the walker in nontrivial ways. To affect the relative phases of $|0\rangle$ and $|1\rangle$, equivalent to a rotation $R_Z(\theta)$ of the encoded qubit by an angle θ about the Z axis, one needs to add an identity widget to the first rail and a phase widget to the second. Both widgets must take the same amount of time to traverse by a continuous-time quantum walk, and both must have PST. For a universal single-qubit gate, one also requires a rotation about an orthogonal axis X or Y. This requires a widget that connects the two rails in such a way that, after continuous evolution for a specified time, the amplitude on the two rails will have been transferred into a different superposition of $|0\rangle$ and $|1\rangle$. This generalizes the concept of PST: arbitrary probability amplitude should remain on the input and output vertices, but no amplitude can remain on any other vertex of the widget.

Figure 2 shows all of these elements combined to form a single-qubit gate via hybrid discrete-continuous quantum walk. The graphs $\mathcal{G}_i$ are to be chosen from a universal set of

graphs that we determine in Sec. III. They are such that when the transport rails attached to them are turned on while the walker is at a node with position $x = 3j$, $j \in \mathbb{Z}$, it undergoes PST to the node at $3j + 1$ and is transformed as desired in the process. The choice of the $\mathcal{G}_i$ is algorithm dependent, but once made the graphs remain in place unchanged throughout the following protocol, which requires a level of global control only to switch among three sets of transport segments.

A walker is initialized on the leftmost vertex of the $|0\rangle$ rail, and the solid transport rail segments, labeled by $\mathcal{G}_g$, are enabled. (Note that at any given time only one transport graph—$\mathcal{G}_g$, $\mathcal{G}_r$, or $\mathcal{G}_b$—is enabled, so when one is stated to be on, the others are implied to be off.) After a time t_h, which must be the same for all $\mathcal{G}_i$ to which horizontal rail segments attach, the walker has unit probability to be at $x = 1$. The next step is taken with $\mathcal{G}_r$ enabled for a time $t_m = \pi/2$, moving the walker to $x = 2$. The graphs $\mathcal{G}_i$ for the vertical rail segments are such that after a time t_v, with $\mathcal{G}_g$ enabled again, either the walker remains unchanged or it is transformed into a superposition of the $|0\rangle$ and $|1\rangle$ rails at $x = 2$. In either case, $\mathcal{G}_b$ is the next to be turned on for a time t_m, moving the walker to $x = 3$, possibly spread over two rails. This sequence now repeats: $\mathcal{G}_g$ for t_h, $\mathcal{G}_r$ for t_m, $\mathcal{G}_g$ for t_v, and $\mathcal{G}_b$ for t_m. Each iteration moves the walker three x positions to the right in a time of $t_h + t_v + \pi$, enacting operations upon it along the way. After traversing the whole graph, involving some number of iterations, the state of the walker at the output on the right will be the desired arbitrary single-qubit state $\alpha|0\rangle + \beta|1\rangle$, with $|\alpha|^2 + |\beta|^2 = 1$. The next step is to expand this scheme from single-qubit operations to universal quantum computation, which will follow from universal single-qubit computation plus a two-qubit entangling gate.

To extend our scheme to two qubits, we require four rails. The horizontal portions of the protocol remain the same; however, we now require vertical connections between additional pairs of rails at each step. The graph that allows this can be seen in Fig. 3. There is a new sequence for switching the transport rail sets on and off, but there are still only three distinct segment types required. For N qubits the number of rails required is 2^N and the number of inter-rail connections at a single x value (e.g., $x = 2$ or 3 in Fig. 3) is 2^{N-1}. Note that

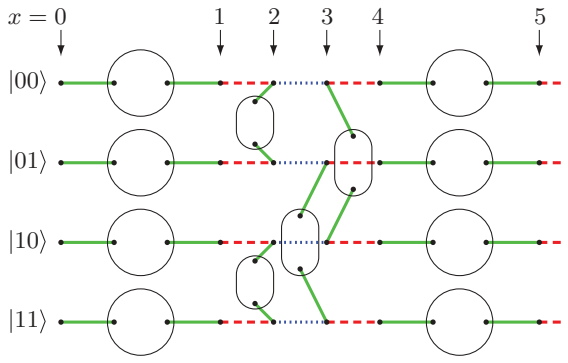

FIG. 3. (Color online) Hybrid scheme for a two-qubit computation. The required sequence for the three transport rail types is g, r, g, b, g, r, repeat. Each additional qubit adds another set of either b, g or r, g before the repeat, such that the $\mathcal{G}_r$ and $\mathcal{G}_b$ graphs alternate on the horizontal rail.

this affects only the width of the graph: the number of distinct x values at which these inter-rail connections are required at each stage in the sequence is only N. This is because, to perform a single-qubit rotation on the nth qubit out of N, the rail corresponding to $|b_1 \cdots 0_n \cdots b_N\rangle$ must be connected to $|b_1 \cdots 1_n \cdots b_N\rangle$ for each of the 2^{N-1} arrangements of the $b_i \neq b_n$, but these connections are simultaneous. Since single-qubit rotations are required for each qubit, we require N such sets of connections at each step, so the depth of the graph and therefore the time taken to traverse it is linear in the number of qubits. The number of these inter-rail connections compares directly with the requirements of previous schemes [15,16].

Given a set of graphs $\{\mathcal{G}_i\}$ that provides a universal gate set, which we describe in the following section, we are now in a position to describe the simulation of a quantum circuit on N qubits, with a depth of D. We define a layer of the computational graph to be one of three things: (1) a horizontal widget on each rail, mediated by $\mathcal{G}_g$, (2) the subgraph of $\mathcal{G}_r$ or $\mathcal{G}_b$ joining x and $x + 1$ for some x, or (3) the subgraph of $\mathcal{G}_g$ at a single x value, providing the vertical connections that allow a basis-changing operation on a single qubit. We further define a horizontal sequence $\mathcal{S}_h$ as the enabling of $\mathcal{G}_g$ for a duration t_h followed by $\mathcal{G}_r$ for t_m, and a vertical sequence $\mathcal{S}_v^{(j)}$, for $j \in \{r,b\}$, as $\mathcal{G}_g$ enabled for t_v followed by $\mathcal{G}_j$ for t_m. The protocol proceeds as follows.

Algorithm-specific graphs are inserted into the generic structure of the transport graphs $\mathcal{G}_g$, $\mathcal{G}_r$, and $\mathcal{G}_b$. With these graphs disabled, a quantum walker is initialized on the first node of the $|00\cdots0\rangle$ rail. The transport graphs are then cycled on and off according to

$$\mathcal{S}_h, \underbrace{\mathcal{S}_v^{(b)}, \mathcal{S}_v^{(r)}, \ldots, \mathcal{S}_v^{(r)}}_{N \text{ sequences}}, \tag{2}$$

with the vertical sequences alternating between r and b. This constitutes the first round of the protocol, and it finishes with the walker at $x = 2(N + 1)$ after a time of $t_h + t_m + N(t_v + t_m)$. The set of sequences in Eq. (2) is executed a total of D times, after which the walker is in the superposition of output nodes corresponding to the result of the action of the circuit unitary on the input state. We can therefore define the "graph depth" by $D_{\mathcal{G}} = 2D(N + 1)$, corresponding to the total number of operations required to simulate the circuit of depth D. Since the graph depth is polynomial in the number of qubits, its dependence on N results in at worst a logarithmic correction factor to any quantum algorithm offering a polynomial speedup over the classical case. Algorithms offering exponential speedups continue to do so in this model.

If additional runs of the algorithm are required, for example to build up statistics of the output state, they can be run almost in parallel. Once the first walker has reached the input node to the second round at position $x = N + 2$, a second walker can be started at the input node of the first round, $x = 0$. With no additional cost, the same sequence of transport segments then moves both walkers through the computation simultaneously, and neither is affected by the other's presence. When the first walker reaches the set of final output nodes, it remains there at the final x position while the last set of transport rails that it traversed is off. During this time it can be measured and

ejected from the system before those rails cycle on again. This prevents the first walker from moving backward into the graph toward the second one. The whole process can of course be repeated for additional walkers.

The distance between walkers can in fact be made constant if the widgets are chosen such that $t_h = t_v$ and the number of vertical sequences required in Eq. (2) is odd, say $N = 2k + 1$, $k \in \mathbb{Z}$.[1] In this case, $\mathcal{S}_h = \mathcal{S}_v^{(r)}$, and the result of Eq. (2) for one round is simply $k + 1$ copies of the sequence $\mathcal{S}_v^{(r)}\mathcal{S}_v^{(b)}$. Therefore, a second walker can be initialized at $x = 0$ after the first of these (i.e., when the first walker has reached $x = 4$, independent of N). For example, with $N = 3$ qubits the sequence for two walkers is

$$\begin{array}{ll} \text{Walker 1} \rightarrow & \overbrace{\qquad\qquad}^{\text{Round 1}}\ \overbrace{\qquad\qquad}^{\text{Round 2}} \\ \text{Graphs:} & g,r,g,b,\underbrace{g,r,g,b}\,,\overbrace{g,r,g,b},\cdots \\ \text{Walker 2} \rightarrow & \qquad\qquad\ \underbrace{\qquad\qquad}_{\text{Round 1}} \end{array} . \tag{3}$$

Our universal gate set, described in the next section, is of this form with $t_h = t_v$.

III. WIDGETS

The fundamental elements of this scheme are the graphs $\mathcal{G}_i$ that are connected to the rails. In this section we describe three widgets, each comprised of a graph attached to two transport rail segments, that together yield a universal gate set for single-qubit operations. When combined, these yield the identity gate, a Z rotation, and an X rotation on the encoded qubit.

In principle, the identity gate is already built into the motion along each rail: the state of the walker after each step is simply multiplied by a factor of $-\mathrm{i}$. The N-qubit state being represented is then unaffected by the motion along each step, besides an unimportant overall phase. That said, all the widgets shown in Fig. 2 are graphs $\mathcal{G}_i$ *combined* with two edges, connecting to the input and output vertices of the $\mathcal{G}_i$, respectively. The smallest identity graph possible is therefore a single vertex, which corresponds to a three-site unweighted linear widget. The same widget can be obtained by dividing the edge weights of the two-segment ($M = 2$) quantum wire [cf. Fig. 1(a)] by a factor of $\sqrt{2}$. Thus, the simplest identity gate requires a PST time $t = \sqrt{2}(\pi/2) = \pi/\sqrt{2}$ and multiplies the state of the walker by a factor of $(-\mathrm{i})^2 = -1$. The same procedure can be applied to quantum wires of arbitrary length: dividing the edge weights by $\sqrt{M}$ yields an identity widget with unit weights on the first and last edges, in a PST time $t = \sqrt{M}(\pi/2)$ and with an overall factor of $(-\mathrm{i})^M = e^{-\mathrm{i}M\pi/2}$ for each rail. Note though that two wires of different length cannot be combined to create a phase gate, since they require different times to exhibit PST.

The first nontrivial gate is a Z rotation, $R_Z(\theta) \propto |0\rangle\langle 0| + e^{\mathrm{i}\theta}|1\rangle\langle 1|$. After a single step of the discontinuous quantum walk, the state of the walker on the second rail (encoding the $|1\rangle$ component of the computational qubit) must accumulate a phase different by θ from that accumulated on the first rail. In practice, this means that the state on the second rail must obtain a phase $\theta \neq -M\pi/2$ in a time $t = \sqrt{M}(\pi/2)$, relative to the identity gate acting on the first rail.

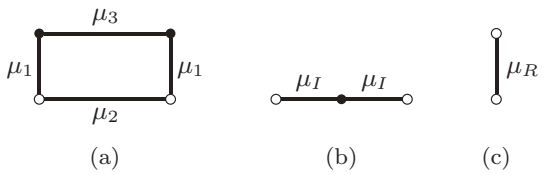

FIG. 4. Basic building blocks of the hybrid scheme. Open circles represent the nodes shared with the transport graph $\mathcal{G}_g$. The graphs are (a) the phase graph $\mathcal{G}_P$, (b) the identity graph $\mathcal{G}_I$, and (c) the rotation graph $\mathcal{G}_R$. Their edge weights are $\mu_I = \sqrt{3/2}$, $\mu_1 = 5\sqrt{3}/8$, $\mu_2 = 15/8$, $\mu_3 = 21/8$, and $\mu_R = 2\sqrt{3}$.

Candidate graphs on up to four vertices were considered, and within this restricted search space no widgets satisfied the above criteria for transit times $t = \sqrt{2}\pi/2$ or $\sqrt{3}\pi/2$. The first successful widget found has transit time $t = \pi$. This is based on the graph $\mathcal{G}_P$ that is a weighted square, as shown in Fig. 4(a). We number the vertices clockwise around the graph, from $|1\rangle$ in the bottom left to $|4\rangle$ in the bottom right; the corresponding widget has two additional vertices, $|v_\ell\rangle$ and $|v_r\rangle$, attached to $\mathcal{G}_P$ by transport rails on the left and right, respectively. The resulting widget Hamiltonian (adjacency matrix) is

$$\begin{aligned} H_P = {} & |v_\ell\rangle\langle 1| + \mu_1(|1\rangle\langle 2| + |3\rangle\langle 4|) + \mu_2|1\rangle\langle 4| \\ & + \mu_3|2\rangle\langle 3| + |4\rangle\langle v_r| + \text{H.c.} \end{aligned} \tag{4}$$

With the edge weightings $\mu_1 \equiv w_{12} = w_{34} = 5\sqrt{3}/8$, $\mu_2 \equiv w_{14} = 15/8$, and $\mu_3 \equiv w_{23} = 21/8$, the state of a walker initially on the left-hand node $|v_\ell\rangle$ is transformed in a time π as

$$e^{-\mathrm{i}H_P\pi}|v_\ell\rangle = \mathrm{i}|v_r\rangle. \tag{5}$$

The time for the horizontal segments of the computational graph is then taken to be $t_h = \pi$.

To perform a Z rotation based on the graph $\mathcal{G}_P$, one requires an identity gate taking a time $t_h = \pi = \sqrt{4}\pi/2$ on the first rail. Evidently this corresponds to a four-segment ($M = 4$) quantum-wire widget, with the weights of the first and last segments rescaled to unity. The phase acquired by the walker during traversal is $(-\mathrm{i})^4 = 1$. This results from the graph $\mathcal{G}_I$, shown in Fig. 4(b), connected to a rail on either end. Since there are four line segments in total, the weighting of the second and third segments should be $\mu_I = \sqrt{3/2}$. The Hamiltonian corresponding to this widget is

$$H_I = |v_\ell\rangle\langle 1| + \mu_I \sum_{i=1}^{2} |i\rangle\langle i+1| + |3\rangle\langle v_r| + \text{H.c.}, \tag{6}$$

where $\{|1\rangle, |2\rangle, |3\rangle\}$ is the set of nodes in $\mathcal{G}_I$, labeled from left to right.

The final graph we require does not actually exhibit PST. Instead, with $\mathcal{G}_g$ enabled, the rotation graph $\mathcal{G}_R$ results in a widget connecting a vertex $|v_t\rangle$ on the top to $|v_b\rangle$ on the bottom of a pair of rails. In a time $t_v = \pi$ the effect of this widget on a walker starting at either $|v_t\rangle$ or $|v_b\rangle$ is to split its probability density between these two states, leaving no probability inside

[1] As we have described the protocol, an odd number of vertical sequences corresponds to an odd number of qubits. If an even number of qubits is desired along with a constant distance between walkers, an additional vertical identity sequence could simply be added.

the graph itself. $\mathcal{G}_R$ consists of a single weighted line segment and can be seen in Fig. 4(c). Its widget Hamiltonian is

$$H_R = |v_t\rangle\langle 1| + \mu_R|1\rangle\langle 2| + |2\rangle\langle v_b| + \text{H.c.}, \tag{7}$$

and, with weighting $\mu_R = 2\sqrt{3}$, the action on walkers initially on either the top or the bottom rail is given by

$$\begin{aligned} e^{-iH_R t_v}|v_t\rangle &= \cos(\sqrt{3}\pi)|v_t\rangle - i\sin(\sqrt{3}\pi)|v_b\rangle, \\ e^{-iH_R t_v}|v_b\rangle &= -i\sin(\sqrt{3}\pi)|v_t\rangle + \cos(\sqrt{3}\pi)|v_b\rangle. \end{aligned} \tag{8}$$

Note that if the graph $\mathcal{G}_R$ is not present, which is equivalent to setting the weight $\mu_R = 0$, then this same widget acts as an identity operation in the same time t_v. In this case, whether it starts at $|v_t\rangle$ or $|v_b\rangle$, the walker sees only a single line segment. It walks the line in time $t_v/2$, acquiring a phase of $-i$. Therefore, after an elapsed time of t_v the walker has made a round trip and returned to its initial position with an accumulated phase of $(-i)^2 = -1$; its state is unchanged, up to a global phase.

IV. UNIVERSAL COMPUTATION

We now show how the graphs $\mathcal{G}_I$, $\mathcal{G}_P$, and $\mathcal{G}_R$ can be combined to construct a universal set of gates for single-qubit operations, and then we add a controlled-Z (CZ) gate to provide universal quantum computation.

The $\sqrt{Z}$ phase gate is straightforward to construct. Consider only the first stage of the graph in Fig. 2, comprised of the graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ along with the connector segments that attach them to the $x = 0$ and $x = 1$ nodes. As shown in Fig. 5, we replace $\mathcal{G}_1$ with the identity graph $\mathcal{G}_I$ and substitute the phase graph $\mathcal{G}_P$ for $\mathcal{G}_2$. A walker starting in an arbitrary superposition of computational basis states on the leftmost vertices of the gate, $|\psi(t=0)\rangle = \alpha|0\rangle + \beta|1\rangle$, will after a time t_h be in the state $|\psi(t=t_h)\rangle = \alpha|0\rangle + \beta e^{i\pi/2}|1\rangle = \sqrt{Z}|\psi(t=0)\rangle$ on the rightmost vertices. Note that placing $\mathcal{G}_P$ on the $|0\rangle$ rail and $\mathcal{G}_I$ on the $|1\rangle$ rail changes the resulting gate into $i\sqrt{Z}^\dagger$, and of course putting $\mathcal{G}_I$ on both rails results in the identity gate, I.

After enacting $\sqrt{Z}$, $i\sqrt{Z}^\dagger$, or I, the walker has moved from $x = 0$ to $x = 1$. It is then transferred to $x = 2$ via PST across a single line segment. At this point a basis change can be effected, if desired, by using the rotation graph $\mathcal{G}_R$ with the states $|v_{\text{in}}\rangle$ and $|v_{\text{out}}\rangle$ of Fig. 4(c) identified with the logical rail states $|0\rangle$ and $|1\rangle$, respectively. Using Eq. (8), one readily obtains

$$|\psi(t=t_v)\rangle = R_X(2\sqrt{3}\pi)|\psi(t=0)\rangle. \tag{9}$$

The $\sqrt{Z}$ and $R_X(2\sqrt{3}\pi)$ gates constitute a universal set for single-qubit operations. Conjugating the latter by the former

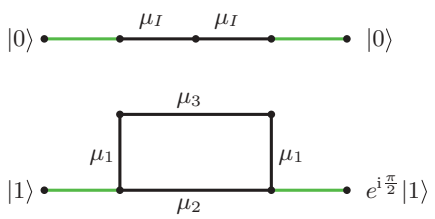

FIG. 5. (Color online) Single-qubit $\sqrt{Z}$ gate.

gives

$$\sqrt{Z}R_X(2\sqrt{3}\pi)\sqrt{Z}^\dagger = R_Y(2\sqrt{3}\pi), \tag{10}$$

that is, a Y rotation through an angle $2\sqrt{3}\pi$. As with Euler-angle rotations in three-dimensional Cartesian space and due to the correspondence between SO(3) and SU(2), these rotations of the Bloch sphere by irrational multiples of π about nonparallel axes allow an arbitrary rotation to be performed and, therefore, provide a universal set of single-qubit gates.

All that remains is to construct a two-qubit entangling gate. The implementation of $\sqrt{Z}$, by the application of a phase to a single computational basis state, suggests a straightforward method for implementing the entangling controlled-$\sqrt{Z}$ gate, $\sqrt{\text{CZ}}$. With rails for two qubits we simply apply $\sqrt{Z}$ to the $|11\rangle$ rail while applying identity operations to the $|00\rangle$, $|01\rangle$, and $|10\rangle$ rails, thus applying a phase of $e^{i\pi/2}$ to $|11\rangle$ relative to the other three computational basis states. Repeating this obviously results in a full CZ operation. In combination with the universal set of single-qubit operations already described, the ability to implement a CZ gate makes this scheme universal for quantum computation.

V. CONCLUSIONS

By combining components of perfect state transfer and quantum walks we have developed a hybrid scheme for performing universal quantum computation via a walker taking discrete steps of continuous evolution, a discontinuous quantum walk. The computational model is based on one rail per computational basis state, as developed for prior schemes to provide universal quantum computation in the distinct cases of continuous [15] and discrete [16] walks. As in the discrete case, we have eliminated the need for the excess tails used in the continuous case to support well-defined momentum states, and we do not require the momentum filter that prevents most of the walker's probability from participating in the computation. By making use of perfect state transfer, we ensure that the walker completes the quantum computation with certainty. Unlike the discrete case, the evolution of our quantum walker is manifestly physical under a specific Hamiltonian, and we do not require site-dependent coins of multiple dimensions or indeed any coin at all. The cost associated with these improvements is an additional amount of global control, which is analogous to the coin and shift operations employed by discrete-time quantum walks with site-independent coins. The required control is algorithm independent, conforming to a well-defined, preprogrammed sequence.

The widgets described in Sec. III are universal for quantum computation, so they provide a proof-of-principle scheme for the implementation of arbitrary quantum algorithms. That said, they are neither unique nor or they likely to be a preferred set for particular applications. Alternative choices of single- and double-rail graphs (generating single-qubit gates) might generate particular desired gates (such as the Hadamard or $\pi/8$ gate) more readily. Multiqubit gates (such as the three-qubit Toffoli gate) could be found by graphs linking multiple rails. A desired unitary for n qubits would conceivably have a

more efficient decomposition in terms of a larger widget set. This decomposition would be in the same spirit as the model employed in Ref. [24] to examine the relationship between discrete and continuous quantum query algorithms, but would require neither so-called fractional queries nor Trotter-Suzuki-type approximations.

Regardless of the choice of widgets, one can recast the Hamiltonians in terms of spin networks, in the spirit of Ref. [25], perhaps providing a closer link to potential experimental implementations. This is possible because a quantum walker on a k-vertex graph can be mapped onto the single-excitation subspace of a system of k spin-1/2 particles under the XY model. The spin-preserving Hamiltonian of this model is of the form $H \sim \frac{1}{2}\sum_i (X_i X_{i+1} + Y_i Y_{i+1})$. In this context the particles themselves remain stationary and take the place of the nodes, while the exchange interaction provides edges along which the excitation propagates. The correspondence between the XY and quantum-walk models can be seen directly in the behavior of two interacting spins: an excitation on the left spin evolves to an excitation on the right one. This is nothing but a Pauli X operation, as effected on a quantum walker under the influence of the hopping Hamiltonian on the two-vertex connected graph.

More generally, the discontinuous quantum walk provides a framework for universal control of a quantum system. Though the universality of quantum computation is presented above in analogy to the circuit model, with rails corresponding to computational basis states, this is not in fact essential. In principle, the edges between subgraphs of any particular graph can be turned on and off in a prescribed manner, in the process effecting some desired operation on the quantum walker. The total number of vertices would still presumably scale exponentially in the number of simulated qubits, but the representation of the graph for some quantum algorithms could be much more efficient than that proposed above. We hope that the flexibility of the discontinuous quantum walk will lend itself naturally to the development of new efficient quantum algorithms.

ACKNOWLEDGMENTS

We are grateful to Andrew Childs, Barry Sanders, and Peter Høyer for fruitful discussions during the preparation of this work. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Alberta Informatics Circle of Research Excellence (iCORE).

[1] G. J. Woeginger, *Combinatorial Optimization—Eureka, You Shrink!* (Springer-Verlag, New York, 2003), pp. 185–207.
[2] J. Kempe, Contemp. Phys. **44**, 307 (2003).
[3] A. Ambainis, Int. J. Quantum Inform. **1**, 507 (2003).
[4] N. Shenvi, J. Kempe, and K. Birgitta Whaley, Phys. Rev. A **67**, 052307 (2003).
[5] M. Santha, Lect. Notes Comput. Sci. **4978**, 31 (2008).
[6] H. Buhrman, R. de Wolf, C. Dürr, M. Heiligman, P. Høyer, F. Magniez, and M. Santha, in *CCC '01: Proceedings of the 16th Annual Conference on Computational Complexity* (IEEE Computer Society, Washington, DC, 2001), pp. 131–137.
[7] A. Ambainis, SIAM J. Comput. **37**, 210 (2007).
[8] A. M. Childs and J. M. Eisenberg, Quantum Inf. Comput. **5**, 593 (2005).
[9] A. M. Childs, E. Farhi, and S. Gutmann, Quantum Inf. Process. **1**, 35 (2002).
[10] A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, and D. A. Spielman, in *STOC '03: Proceedings of the 35th Annual ACM Symposium on Theory of Computing* (ACM, New York, 2003), pp. 59–68.
[11] E. Farhi and S. Gutmann, Phys. Rev. A **58**, 915 (1998).
[12] E. Farhi, J. Goldstone, and S. Gutmann, Theory Comput. **4**, 169 (2008).
[13] A. M. Childs, R. Cleve, S. P. Jordan, and D. Yonge-Mallo, Theory Comput. **5**, 119 (2009).
[14] B. W. Reichardt, e-print arXiv:0907.1623.
[15] A. M. Childs, Phys. Rev. Lett. **102**, 180501 (2009).
[16] N. B. Lovett, S. Cooper, M. Everitt, M. Trevers, and V. Kendon, Phys. Rev. A **81**, 042330 (2010).
[17] S. Bose, Contemp. Phys. **48**, 13 (2005).
[18] A. Kay, Int. J. Quantum Inform. **8**, 641 (2010).
[19] M. Christandl, N. Datta, A. Ekert, and A. J. Landahl, Phys. Rev. Lett. **92**, 187902 (2004).
[20] D. L. Feder, Phys. Rev. Lett. **97**, 180502 (2006).
[21] A. Bernasconi, C. Godsil, and S. Severini, Phys. Rev. A **78**, 052320 (2008).
[22] S. Severini, e-print arXiv:1001.0674.
[23] C. Godsil and S. Severini, Phys. Rev. A **81**, 052316 (2010).
[24] R. Cleve, D. Gottesman, M. Mosca, R. D. Somma, and D. Yonge-Mallo, in *STOC '09: Proceedings of the 41th Annual ACM Symposium on Theory of Computing* (ACM, New York, 2009), pp. 409–416.
[25] M. Christandl, N. Datta, T. C. Dorlas, A. Ekert, A. Kay, and A. J. Landahl, Phys. Rev. A **71**, 032312 (2005).