QWalk: Simulador de Caminhadas Quânticas

F.L. Marquezino[1], **R. Portugal**[1]

[1]Laboratório Nacional de Computação Científica, LNCC
Av. Getúlio Vargas, 333 – CEP 25651075 – Petrópolis – RJ – Brasil

{franklin,portugal}@lncc.br

Abstract. *Since random walks have been used with great success in the development of classical algorithms, it is expected that similar techniques could be used in quantum computation in order to obtain new efficient algorithms. Hence, several research groups are giving special attention to quantum walks recently. In fact, new algorithms using quantum walks have already been developed. Despite all the important results and perspectives of this field, there is no quantum walk simulator available to the scientific community yet. In this paper we present QWalk, a quantum walk simulator for one- and bidimensional lattices with broken links. We use examples to explain the usage of the software and to show some recent results of literature that are easily reproduced by the simulator.*

Resumo. *Como caminhadas aleatórias têm sido utilizadas com grande sucesso no desenvolvimento de algoritmos clássicos, espera-se que técnicas semelhantes possam ser utilizadas na Computação Quântica a fim de obter novos algoritmos eficientes. Portanto, diversos grupos de pesquisa estão prestando especial atenção às caminhadas quânticas recentemente. De fato, novos algoritmos já foram desenvolvidos utilizando caminhadas quânticas. Apesar de todos os importantes resultados e perspectivas desta área, não há ainda simulador de caminhadas quânticas disponível para a comunidade científica. Neste artigo apresentamos QWalk, um simulador de caminhadas quânticas para redes uni- e bidimensionais com ligações interrompidas. Usamos exemplos para explicar o uso do* software *e para mostrar alguns resultados recentes da literatura que são facilmente reproduzidos pelo simulador.*

1. Introdução

Uma abordagem utilizada com muito sucesso na Computação Clássica é a de caminhadas aleatórias — também conhecidas como passeios aleatórios. Diversos problemas computacionais podem ser resolvidos mais eficientemente através desta técnica [Motwani and Raghavan 1995]. Na década de 1990 foi desenvolvido inicialmente o modelo discreto de caminhadas quânticas [Aharonov et al. 1993], e em seguida o modelo contínuo. Motivados por estas descobertas podemos investigar a utilização de caminhadas quânticas para o desenvolvimento de novos algoritmos, na esperança de que assim como esta técnica proporcionou bons resultados na Computação Clássica, sua correspondente quântica também possa trazer algoritmos eficientes para a Computação Quântica.

Alguns algoritmos quânticos já foram desenvolvidos fazendo uso da técnica de caminhadas quânticas. Como exemplos, pode-se mencionar o algoritmo de busca de [Shenvi et al. 2003] e o algoritmo para distinção de elementos de [Ambainis 2004].

Além das aplicações para o desenvolvimento de algoritmos quânticos eficientes, importantes do ponto de vista da Ciência da Computação, as caminhadas quânticas possuem propriedades que por si próprias já justificariam seu estudo do ponto de vista da Física. Uma técnica promissora neste sentido é a de ligações interrompidas, desenvolvida inicialmente por [Romanelli et al. 2005] e posteriormente generalizada para o caso bidimensional por [Oliveira et al. 2006a].

Nota-se que até hoje não se encontra disponível um simulador genérico de caminhadas quânticas. Isto faz com que os esforços de pesquisadores sejam desviados para a implementação de simulações específicas, enquanto deveriam estar concentrados nos aspectos físicos e matemáticos da pesquisa. O simulador QWalk, apresentado neste trabalho, preenche esta lacuna e possibilita aos pesquisadores a realização de simulações importantes de modo simplificado.

Na Seç. 2 revisamos rapidamente as caminhadas aleatórias quânticas discretas. Na Seç. 3 apresentamos o simulador QWalk e destacamos alguns de seus aspectos técnicos. Na Seç. 4 demonstramos o uso do simulador e exemplificamos sua versatilidade reproduzindo alguns resultados recentes da literatura. No Apêndice, descrevemos algumas opções do simulador que não foram contempladas pelos exemplos.

2. Caminhadas aleatórias quânticas

Na caminhada quântica unidimensional temos uma partícula livre que se movimenta a cada instante sobre uma rede unidimensional infinita. O sentido do movimento é dado por um grau de liberdade adicional da própria partícula, correspondendo, no caso clássico, ao resultado do lançamento de uma moeda. Na caminhada quântica bidimensional, de modo bastante semelhante, tomamos uma rede bidimensional infinita e dois graus de liberdade para a moeda, possibilitando decidir entre os quatro tipos de movimento. Nesta Seção descrevemos o caso bidimensional. A caminhada unidimensional, no entanto, pode ser vista como uma simples particularização do caso aqui descrito.

O espaço de Hilbert considerado na caminhada bidimensional é $\mathcal{H}_4 \otimes \mathcal{H}_\infty$, em que $\mathcal{H}_4$ é o espaço da moeda e $\mathcal{H}_\infty$ é o espaço da posição da partícula. A base para o espaço da moeda é $\{|j,k\rangle : j,k \in \{0,1\}\}$, enquanto a base para o espaço da posição é $\{|m,n\rangle : m,n \in \mathbb{Z}, m+n \text{ par}\}$.

O estado genérico do caminhante quântico é dado por

$$|\psi(t)\rangle = \sum_{j,k=0}^{1} \sum_{m,n=-\infty}^{\infty} A_{j,k;m,n}(t) \, |j,k\rangle \, |m,n\rangle. \tag{1}$$

A evolução temporal do sistema é dada pelo operador $U = S \circ (C \otimes I_P)$, em que C é o operador moeda, I_P é a identidade atuando no espaço da posição, e

$$S = \sum_{j,k=0}^{1} \sum_{m,n=-\infty}^{+\infty} |j,k\rangle \langle j,k| \otimes \left|m + (-1)^j, n + (-1)^k\right\rangle \langle m,n| \tag{2}$$

é o operador de translação.

O operador moeda pode ser qualquer operador unitário atuando no espaço $\mathcal{H}_4$. Sua função é modificar o estado da moeda quântica em cada iteração. O operador translação

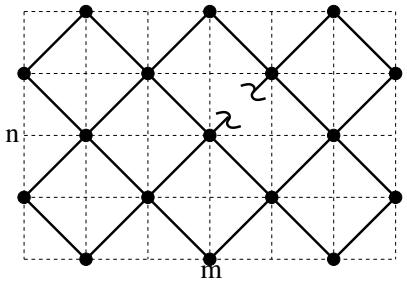


Figura 1. Parte da rede para caminhada quântica bidimensional, exibindo uma ligação interrompida.

descreve o movimento da partícula na rede. Esta, anda um passo na diagonal principal se a moeda for $|0,0\rangle$ ou $|1,1\rangle$ e anda um passo na diagonal secundária se a moeda for $|0,1\rangle$ ou $|1,0\rangle$.

Podemos considerar que, em um certo instante, o sítio (m,n) possua alguma ligação para sítios vizinhos interrompida [Romanelli et al. 2005]. Para isso, precisamos definir as funções

$$\mathcal{L}_1(j,k;m,n) = \begin{cases} (-1)^j & \text{se ligação para sítio } m+(-1)^j, n+(-1)^k \text{ está fechada,} \\ 0 & \text{se ligação está aberta,} \end{cases} \tag{3}$$

$$\mathcal{L}_2(j,k;m,n) = \begin{cases} (-1)^k & \text{se ligação para sítio } m+(-1)^j, n+(-1)^k \text{ está fechada,} \\ 0 & \text{se ligação está aberta,} \end{cases} \tag{4}$$

em que $j,k \in \{0,1\}$. Sempre que $\mathcal{L}_1(1-j,1-k;m+(-1)^j,n+(-1)^k)=0$ temos $\mathcal{L}_1(j,k;m,n)=0$, e analogamente para $\mathcal{L}_2$.

Na Fig. 1 vemos uma parte da rede utilizada na caminhada quântica bidimensional. A rede matemática é indicada por uma linha pontilhada enquanto a rede física, sobre a qual se dá o movimento da partícula, é indicada por uma linha cheia. No exemplo, vemos uma ligação interrompida entre os sítios (m,n) e $(m+1,n+1)$.

Aplicando o operador de evolução temporal a (1), e incluindo as funções $\mathcal{L}_1$ e $\mathcal{L}_2$ como em [Oliveira et al. 2006a], temos então a equação de evolução

$$A_{1-j,1-k;m,n}(t+1) = \sum_{j',k'=0}^{1} C_{j+\mathcal{L}_1(j,k;m,n),k+\mathcal{L}_2(j,k;m,n);j',k'} A_{j',k';m+\mathcal{L}_1(j,k;m,n),n+\mathcal{L}_2(j,k;m,n)}(t). \tag{5}$$

3. Aspectos técnicos do QWalk

O simulador QWalk foi totalmente implementado em linguagem C. O código é compatível com o padrão ISO C99, de modo a proporcionar grande portabilidade. Mesmo compiladores mais antigos podem ser capazes de compilar o código-fonte, desde que suportem os requisitos do padrão ISO C99 relacionados a operações com números complexos. Neste artigo é descrita a versão 0.9 do simulador. O código-fonte desta versão, bem como futuras atualizações, pode ser obtido em `http://www.lncc.br/~franklin/qwalk`.

Os principais arquivos que compõem o simulador são `qw1d.c`, `qw2d.c` e `qwamplify.c`, bem como a biblioteca `libqwalk.a`. O primeiro arquivo implementa a ferramenta para simulação de caminhantes quânticos em redes unidimensionais; o segundo implementa para redes bidimensionais; e o terceiro implementa uma ferramenta para amplificar certas regiões de uma função de onda, facilitando a visualização da mesma. Além destas três ferramentas, o usuário programador pode utilizar a biblioteca `libqwalk.a` para desenvolver outros *softwares* de acordo com sua necessidade.

A biblioteca `libqwalk.a`, também desenvolvida no presente trabalho, consiste de funções presentes em diversos arquivos. Os arquivos `qwmem_<tipo de dados>.c` implementam funções de gerenciamento de memória, tratando estruturas de dados dos tipos complexo, inteiro e ponto flutuante. Os demais arquivos também possuem nomes que esclarecem seus conteúdos. Por exemplo, o arquivo `qwcoin.c` implementa funções relacionadas com a criação das principais moedas utilizadas na simulação. Semelhantemente, o arquivo `qwcoin_io.c` implementa funções de entrada e saída de dados de moeda, como por exemplo a leitura de moeda arbitrária a partir de um arquivo. Descrições detalhadas de cada arquivo e de suas respectivas funções podem ser encontradas em comentários no próprio código-fonte.

4. Uso do QWalk

A instalação do QWalk em sistema operacional Linux ou semelhante é bastante simples. Basta salvar os arquivos em uma pasta qualquer e utilizar o comando `make`. Em sistema operacional Windows foram realizados testes bem-sucedidos de compilação através do compilador Dev-C++ 4.9.9.2, obtido gratuitamente na Internet[1]. Juntamente com os arquivos que acompanham o simulador pode-se encontrar informações mais detalhadas sobre a instalação do mesmo.

O simulador consiste em três softwares: o *qw1d* simula caminhadas aleatórias em rede unidimensional; o *qw2d*, em rede bidimensional; e o *qwamplify* amplifica algumas regiões dos gráficos de *qw2d*, permitindo uma melhor visualização.

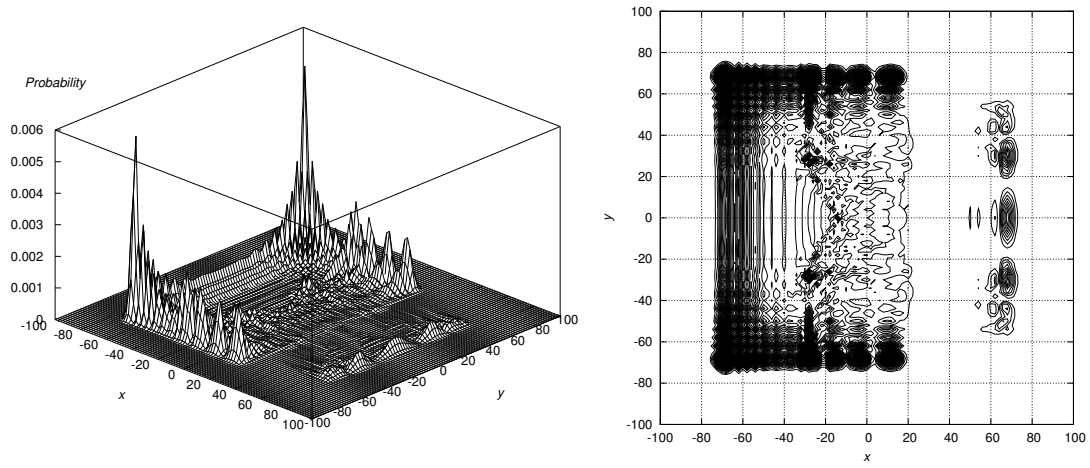
4.1. Simulações bidimensionais

Para utilizar o *qw2d* é necessário um arquivo de entrada, que pode ser gerado em qualquer editor de textos ASCII. Este arquivo de entrada consiste de palavras-chave que definem as opções de simulação. Exemplificamos a seguir o uso das principais palavras-chave.

Na Fig. 2 temos o resultado da simulação de um experimento de fenda dupla com caminhantes quânticos, reproduzindo os resultados obtidos por [Oliveira et al. 2007]. A simulação demorou menos de 2s em um Pentium IV 2.6GHz com 512MB de RAM usando Linux. Para realizar esta simulação no *qw2d*, o arquivo de entrada deve ter as seguintes palavras-chave, em letras maiúsculas:

```
BEGIN
 COIN HADAMARD       BLPERMANENT
 STATE HADAMARD      SCREEN 60 -100 60 100
 STEPS 100
END
```

[1] `http://www.bloodshed.net`.



(a) Gráfico 3D (b) Gráfico de contorno

Figura 2. Experimento da fenda dupla. Um fator de amplificação 5 foi utilizado para $x > 20$, a fim de melhorar a visualização.

Não é necessário agrupar os comandos da mesma forma que o exemplo. O usuário pode, por exemplo, pular linhas entre comandos diferentes, ou até escrever todos na mesma linha, se desejar. É importante, no entanto, que as palavras estejam na seção principal do arquivo de entrada, delimitada entre um comando `BEGIN` e um comando `END`. Caso não estejam, elas são consideradas comentário. Convém ressaltar que os gráficos da Fig. 2 foram gerados com *gnuplot*, através de *scripts* automáticos do *qw2d*. A geração dos gráficos demorou cerca de 30s.

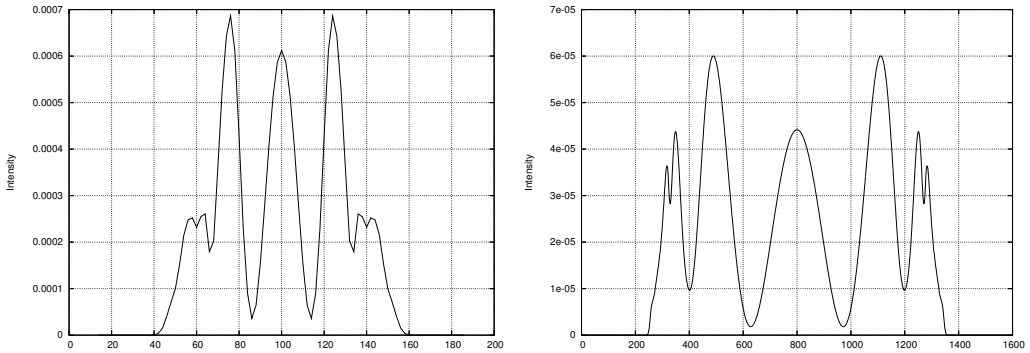
A palavra-chave `COIN` define a moeda utilizada na simulação, que neste caso é a moeda de Hadamard. Poderíamos ter selecionado as moedas de Fourier ou de Grover, por meio das opções `FOURIER` e `GROVER`, ou uma moeda arbitrária, por meio da opção `CUSTOM`.

De maneira análoga, a palavra `STATE` define o estado inicial da simulação. Selecionamos o estado que proporciona o maior espalhamento para a moeda de Hadamard, mas poderíamos ter selecionado os estados correspondentes para a moeda de Fourier e Grover, ou mesmo um estado arbitrário.

A palavra-chave `STEPS` define o número de iterações da simulação. No exemplo, o caminhante dá 100 passos.

O anteparo é definido através da palavra-chave `SCREEN`, que deve ser utilizada na seção principal do arquivo de entrada, seguida das coordenadas inicial e final. O anteparo pode ser posicionado paralelamente aos eixos x ou y, ou com uma inclinação de 45 graus em relação a um destes. No exemplo, o anteparo vai de $(60, -100)$ até $(60, 100)$.

Para criar as fendas utilizamos a palavra-chave `BLPERMANENT` na seção principal do arquivo de entrada. Através deste comando declaramos que a simulação terá ligações interrompidas permanentemente. Para definir a posição destas ligações interrompidas, utilizamos uma seção separada no mesmo arquivo de entrada, com os comandos



(a) Com $T = 100$ e anteparo em $x = 60$ (b) Com $T = 800$ e anteparo em $x = 500$

Figura 3. Simulação de anteparos no experimento da fenda dupla.

```
BEGINBL
 LINE 20 100 20 7
 LINE 20 5   20 -5
 LINE 20 -7  20 -100
ENDBL
```

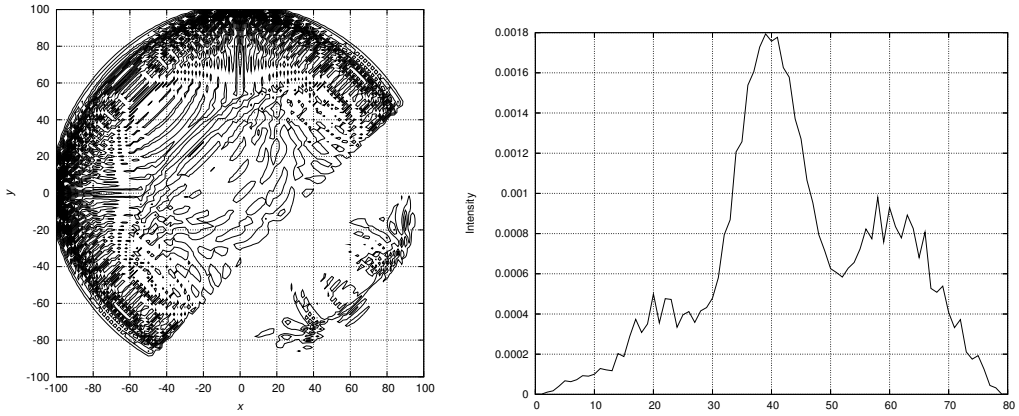
O comando `LINE x0 y0 x1 y1` isola todos os pontos passando sobre o segmento que vai de $(x0, y0)$ até $(x1, y1)$. Também é possível isolar um único ponto, através do comando `POINT x0 y0`.

Os dados gerados inicialmente através do software *qw2d* não permitiam uma boa visualização da parte que atravessava a fenda. Para resolver isso, utilizamos a ferramenta *qwamplify* para amplificar por um fator 5 toda a região em que $x \geq 20$. Para utilizar esta ferramenta, o usuário digita algo como `qwamplify arquivo.dat [opcoes]`. O software salva uma cópia de segurança de `arquivo.dat`, e o substitui por um novo, com parte da função de onda amplificada. Obtém-se ajuda quanto às opções disponíveis digitando apenas `qwamplify`.

Utilizando a opção de ligações interrompidas permanentemente pode-se também reproduzir facilmente os resultados para redes bidimensionais finitas com condições de contorno reflexivas, obtidos em [Oliveira et al. 2006b].

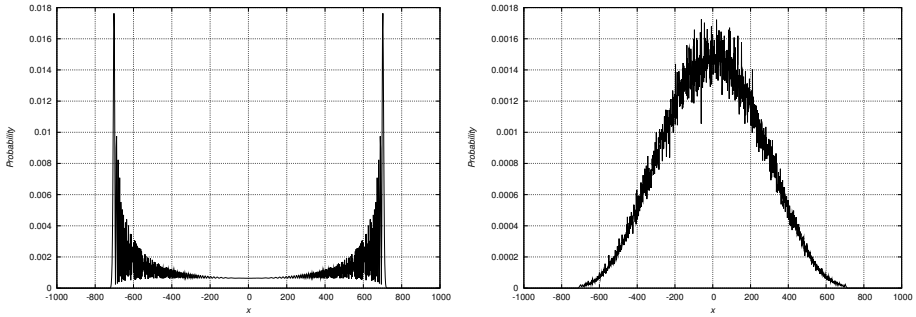
Na Fig. 3 temos, ainda, dois experimentos de fenda dupla com caminhantes de Hadamard. No primeiro foram executados $T = 100$ passos enquanto no segundo, $T = 800$ passos — com simulação levando cerca de 12min no segundo caso. Nestes gráficos vemos os padrões que seriam observados em anteparos colocados, respectivamente, sobre $x = 60$ e $x = 500$. No segundo caso vemos que os mínimos locais decresceram a as curvas tornaram-se mais suaves.

Na Fig. 4 vemos os gráficos de outro experimento de fenda dupla, dessa vez utilizando o caminhante de Grover e a parede posicionada na diagonal. A parede vai de $(-60, -100)$ até $(100, 60)$. Uma fenda vai de $(13, -27)$ até $(15, -25)$ e a outra fenda vai de $(25, -15)$ até $(27, -13)$. Um detector foi posicionado perto da primeira fenda, em $(15, -27)$, e o experimento foi repetido dez vezes. Para definir a posição do detector utilizamos, na seção principal do arquivo de entrada, a palavra-chave `DETECTORS` seguida do número de detectores e suas respectivas coordenadas. Para definir o número de repetições do experimento, utilizamos a palavra-chave `EXPERIMENTS`. A seguir vemos



(a) Gráfico 3D (b) Simulação do anteparo

Figura 4. Experimento da fenda dupla com caminhante de Grover, parede e anteparo na diagonal, e detector próximo a uma das fendas.



(a) Com $T = 1000$ e $p = 0$ (b) Com $T = 1000$ e $p = 0.01$, tomando média de 100 experimentos

Figura 5. Caminhada quântica na reta com ligações interrompidas.

um exemplo de utilização destas duas palavras-chave:

```
DETECTORS 1 15 -27
EXPERIMENTS 10
```

O anteparo, também na diagonal, vai de $(20, -100)$ até $(100, -20)$. Nota-se pelos gráficos que o padrão de interferência foi assimétrico, e menor no lado do detector.

4.2. Simulação unidimensional

Na Fig. 5 vemos a distribuição de probabilidades da caminhada de Hadamard em rede unidimensional, comparando o caso em que a probabilidade de ligações interrompidas é $p = 0$ com o caso em que $p = 0.01$. Em ambos os casos executamos $T = 1000$ passos, utilizando o software *qw1d*. No segundo caso tomamos como resultado a média após 100 experimentos independentes. Nota-se que, devido ao número de passos ter sido muito maior que $1/p$, o comportamento clássico do caminhante aleatório já se tornou bastante evidente, como previsto em [Romanelli et al. 2005].

O uso do *qw1d* é bem semelhante ao uso do *qw2d*, exceto por não haver algumas palavras-chave como `SCREEN`, `BLPERMANENT` e `DETECTORS`, nem certas subopções como `FOURIER` e `GROVER` para moedas e estados. Para definir a probabilidade

de ligações interrompidas em cada passo, utilizamos na seção principal do arquivo de entrada o comando `BLPROB 0.01`, próprio do *qw1d*. Desse modo, em cada iteração, cada ligação tem probabilidade 0.01 de estar interrompida.

5. Conclusões

Neste trabalho, inicialmente, apresentamos sucintamente as caminhadas quânticas em rede infinita bidimensional com possibilidade de ligações interrompidas. As caminhadas unidimensionais são uma particularização do caso apresentado. Apresentamos, então, o simulador QWalk e passamos a descrever seu funcionamento. O QWalk é um simulador de caminhadas quânticas para redes unidimensionais e bidimensionais, compilável em diferentes sistemas operacionais e de fácil utilização.

As principais opções do simulador foram apresentadas através de exemplos. Em primeiro lugar, mostramos o resultado da simulação de um experimento de fenda dupla com o caminhante de Hadamard [Oliveira et al. 2007]. Nos gráficos resultantes observamos claramente os efeitos de difração e interferência, de certo modo semelhantes aos que seriam obtidos em um experimento utilizando fótons. Os resultados foram obtidos com a ferramenta *qw2d* do simulador. Para melhorar a visualização dos resultados foi utilizada a ferramenta de pós-processamento *qwamplify*. Ainda foi possível simular a observação de um anteparo colocado em frente à fenda, a uma certa distância.

Em seguida, analisamos a simulação de outro experimento de fenda dupla. Desta vez, consideramos o caminhante de Grover e colocamos a parede em linha diagonal. Utilizamos fendas ligeiramente maiores e posicionamos um detector próximo a uma das fendas. Desse modo, notamos que o padrão de interferência tornou-se assimétrico, além de ter diminuído consideravelmente no lado do detector.

Finalmente, simulamos um caminhante quântico unidimensional em rede com ligações interrompidas aleatoriamente, segundo uma probabilidade p. Executamos por um número de passos que excedia bastante $1/p$, e notamos o surgimento da caminhada clássica. Este comportamento é o mesmo observado em [Romanelli et al. 2005].

As simulações apresentadas neste trabalho correspondem a trabalhos recentes na área de Caminhadas Quânticas, e foram realizadas de forma bastante simplificada por meio do simulador proposto. O simulador, além de ter reproduzido resultados disponíveis na literatura com bastante precisão, também serve de importante ferramenta para os pesquisadores realizarem novos experimentos.

Uma das principais possibilidades na atual versão do simulador é a investigação do comportamento da caminhada quântica com diferentes fronteiras, por meio da definição adequada de ligações interrompidas. Também pode-se investigar a influência de detectores na caminhada quântica. O simulador também pode auxiliar em uma eventual continuação do estudo realizado por [Oliveira et al. 2007], para fendas simples e duplas, ou mesmo em simulações inspiradas em outros experimentos físicos.

Em versões futuras, pode-se efetuar algumas melhorias no QWalk. A leitura dos dados do arquivo de entrada pode ser melhorada. Por exemplo, se for permitida a utilização de constantes numéricas, o usuário não mais precisará digitar várias casas decimais ao descrever matrizes e estados personalizados. Na versão atual, o simulador calcula média, variância e desvio padrão. Será importante, em versões futuras, o cálculo de es-

tatísticas mais avançadas, como *mixing time* e coeficiente de difusão. O tratamento das ligações interrompidas também pode ser melhorado para facilitar a definição de contornos mais complexos — que não envolvam apenas retas e pontos — e para permitir, por exemplo, a definição de ligações interrompidas dependentes do tempo. Também pode ser útil ao pesquisador que o simulador salve arquivos com a função de onda em instantes intermediários da simulação.

Agradecimentos. Especiais agradecimentos a Amanda Oliveira, Gonzalo Abal e Raul Donangelo, por importantes discussões. O autor FLM conta com bolsa de doutorado do CNPq.

Referências

Aharonov, Y., Davidovich, L., and Zagury, N. (1993). Quantum random walks. *Phys. Rev. A*, 48(2):1687–1690.

Ambainis, A. (2004). Quantum walk algorithm for element distinctness. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*.

Motwani, R. and Raghavan, P. (1995). *Randomized Algorithms*. Cambridge University Press.

Oliveira, A., Portugal, R., and Donangelo, R. (2006a). Decoherence in two-dimensional quantum walks. *Phys. Rev. A*, 74:012312.

Oliveira, A., Portugal, R., and Donangelo, R. (2006b). Two-dimensional quantum walks with boundaries. In *Anais do WECIQ 2006*, pages 211–218. UCPel.

Oliveira, A., Portugal, R., and Donangelo, R. (2007). Simulation of the Single- and Double-Slit Experiments with Quantum Walkers. arXiv:0706.3181v1 [quant-ph].

Romanelli, A., Siri, R., Abal, G., Auyuanet, A., and Donangelo, R. (2005). *Physica A*, 347C(137).

Shenvi, N., Kempe, J., and Whaley, K. B. (2003). Quantum random-walk search algorithm. *Physical Review A*, 67:052307.

A. Mais algumas opções do simulador

Neste Apêndice descrevemos palavras-chave que não foram utilizadas nos exemplos.

`CHECK` faz com que o simulador realize verificações de consistência ao longo da simulação. Pode-se utilizar as sub-opções `STATEPROB` para verificar unitariedade do estado e `SYMMETRY` para verificar simetria[2]. Nas simulações bidimensionais podemos selecionar dois tipos de verificação de simetria. `XSYMMETRY` verifica a simetria em torno do eixo x, ou seja, verifica se as probabilidades nos sítios com (x, y) são iguais às probabilidades nos sítios $(-x, y)$. Analogamente, temos `CHECK YSYMMETRY`.

Para definir uma moeda personalizada, deve-se especificá-la em uma seção à parte do arquivo de entrada. As entradas da matriz devem ser passadas com parte real e parte imaginária. Utilizando, por exemplo, o código

[2]Naturalmente, esta última sub-opção somente pode ser utilizada quando a simetria do estado já for esperada *a priori*.

```
BEGINCOIN
 0.707106781186 0.0    0.0 0.707106781186
 0.0 0.707106781186    0.707106781186 0.0
ENDCOIN
```

definimos, em simulação unidimensional, a moeda

$$C = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & i \\ i & 1 \end{pmatrix}. \tag{6}$$

Para definir um estado personalizado, deve-se especificá-lo em uma seção à parte do arquivo de entrada, descrevendo todas as amplitudes não-nulas. Em simulação unidimensional, o primeiro inteiro refere-se à moeda, o segundo inteiro refere-se à posição do caminhante, e os dois números em ponto flutuante referem-se à parte real e à parte imaginária da amplitude, respectivamente. A palavra-chave `ENDSTATE`, na verdade, serve apenas para organização do usuário, sendo ignorada pelo software. Encerra-se a descrição do estado através do número -1. Utilizando, por exemplo, o código

```
BEGINSTATE
 1 0 0.0 1.0
 -1
ENDSTATE
```

definimos o estado inicial

$$|\psi_0\rangle = i|1\rangle|0\rangle. \tag{7}$$

Em simulações bidimensionais a descrição do estado é análoga, apenas utilizando dois inteiros para moeda e dois para posição do caminhante.

`SEED` define manualmente uma semente para o gerador de números aleatórios. O *default* é um número obtido do relógio do sistema, e normalmente o usuário não deve alterar esta opção.

Em simulações bidimensionais, `AFTERMEASURE` define o número de iterações que serão executadas após uma medição não-trivial, ou seja, após o resultado de uma medição corresponder a um dos detectores em vez do complemento.

`LATTSIZE` e `LATTEXTRA` são usadas mais raramente. A primeira define o tamanho da rede, e seu *default* é 100. Consideramos que a rede vai de $x = -max$ até $x = max$, em que max é o inteiro passado pela palavra-chave `LATTSIZE`. Já a segunda palavra-chave define um espaço extra, a ser reservado para a rede, a fim de evitar acesso a regiões inválidas de memória durante a simulação. Seu valor *default* é 1, e este valor normalmente não deve ser alterado pelo usuário. A ordem correta de utilização é `LATTEXTRA` depois `STEPS`, e finalmente `LATTSIZE`, quando estas palavras-chave são utilizadas simultaneamente.