Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Ciência da Computação

Ataques Quânticos a Geradores de Números Pseudo-Aleatórios

Elloá Barreto Guedes da Costa

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande – Campus I como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Ciência da Computação
Linhas de Pesquisa: Metodologia e Técnicas da Computação, Computação Quântica

Francisco Marcos de Assis e Bernardo Lula Jr.
(Orientadores)

Campina Grande – Paraíba – Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

C837d Costa, Elloá Barreto Guedes da.
Ataques Quânticos a Geradores de Números Pseudo-Aleatórios / Elloá Barreto Guedes da Costa. ─ Campina Grande, 2011.
140 f.: il.

Dissertação (Mestrado em Ciência da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.
Orientadores: Prof. Dr. Francisco Marcos de Assis, Prof. Dr. Bernardo Lula Júnior.
Referências.

1. Geradores Pseudo-Aleatórios. 2. Algoritmos Quânticos. 3. Ataques Criptoanalíticos. 4. Computação Quântica I. Título.

CDU – 004 (043)

"ATAQUES QUÂNTICOS A GERADORES DE NÚMEROS PSEUDO-ALEATÓRIOS"

ELLOÁ BARRETO GUEDES DA COSTA

DISSERTAÇÃO APROVADA EM 25.03.2011

FRANCISCO MARCOS DE ASSIS, Dr.
Orientador(a)

BERNARDO LULA JUNIOR, Dr.
Orientador(a)

AERCIO FERREIRA DE LIMA, Dr.
Examinador(a)

EDMAR CANDEIA GURJÃO, D.Sc
Examinador(a)

VALDEMAR CARDOSO DA ROCHA JUNIOR, Dr.
Examinador(a)

CAMPINA GRANDE - PB

Resumo

Este trabalho apresenta um ataque quântico de comprometimento permanente ao gerador pseudo-aleatório de Blum-Micali. A segurança deste gerador, classificado como criptograficamente seguro, baseia-se na hipótese de intratabilidade do problema do logaritmo discreto perante a Computação Clássica. O ataque proposto faz uso do algoritmo quântico de busca em conjunto com o algoritmo quântico para o logaritmo discreto para comprometer a imprevisibilidade do gerador, recuperando todas as saídas passadas e futuras do mesmo. O presente trabalho também descreve generalizações deste ataque que o adequam a uma gama mais vasta de geradores, incluindo geradores da Construção de Blum-Micali e geradores com múltiplos predicados difíceis. Tais generalizações também abrangem a realização de ataques em situações adversas, por exemplo, quando o adversário captura bits não consecutivos ou quando há menos bits que o requerido. Comparado à sua contrapartida clássica, o algoritmo quântico proposto nesse trabalho possui um ganho quadrático em relação à recuperação do representante do estado interno do gerador, seguido de um ganho superpolinomial na obtenção dos demais elementos do estado interno. Estes resultados caracterizam ameaças, elaboradas com Computação Quântica, contra a segurança de geradores utilizados em diversas aplicações criptográficas.

Palavras-Chave. Geradores Pseudo-Aleatórios, Algoritmos Quânticos, Ataques Criptoanalíticos, Computação Quântica.

Abstract

This dissertation presents a quantum permanent compromise attack to the Blum-Micali pseudorandom generator. The security of this generator, classified as cryptographically secure, is based on the hypothesis of intractability of the discrete logarithm problem in Classical Computing. The proposed attack is based on the quantum search algorithm jointly with the quantum discrete logarithm procedure and aims to compromise the unpredictability of the referred generator, recovering all of its past and future outputs. This work also describes generalizations that enables attacks to generators from the Blum-Micali construction and also to generators with multiple hard-core predicates. Such generalizations also allow attacks when the adversary intercepts non-consecutive bits or when there are less bits than required. Compared to its classical counterpart, the proposed algorithm has a quadractic speedup regarding the retrieval of the representant of the generator's internal state followed by a superpolynomial speedup regarding the obtention of the entire generator's internal state. These results represent menaces of the Quantum Computing paradigm against the security of pseudorandom generators adopted in many real-world cryptosystems.

Keywords. Pseudorandom Generators, Quantum Algorithms, Cryptanalytic Attacks, Quantum Computing.

Agradecimentos

É com grande alegria que estou prestes a celebrar mais uma conquista em minha vida. Têm sido um percurso longo, cheio de desafios, mas cujo traçado está repleto de pessoas bastante generosas, que sempre me motivaram a seguir adiante.

Em primeiro lugar, gostaria de agradecer a Deus pela iluminação e proteção divinas, sempre presentes. Muitos agradecimentos aos meus pais e irmãos, grandes incentivadores do meu trabalho, sempre me apoiando em todos os momentos e compreendendo a minha ausência.

Aos meus orientadores Profs. Francisco Marcos de Assis e Bernardo Lula Jr. que sempre tiveram paciência diante das minhas limitações, generosidade para me fazer crescer intelectualmente, e dedicação no processo de orientação do meu trabalho. Sou muito grata por tudo o que aprendi durante os diversos trabalhos que temos desenvolvido. Que venham outros!

A minha querida Andréa Mendonça, que sempre teve palavras e gestos cordiais nos momentos difíceis e que sempre me inspirou a ser perseverante. É um prazer compartilhar fraternalmente mais esta conquista contigo.

Ao meu querido Francisco Neto, amigo desde a infância. Meu muito obrigada por sempre ter podido contar contigo durante todos estes anos. Quero sempre poder desfrutar de sua amizade!

Um grande agradecimento aos meus amigos do IQuanta por sempre terem sido uma acolhedora família para mim: Christiane Lemos, Eline Alves, Gilson Santos, Heron Aragão, M. Vinícius Rodrigues e os Profs. Aércio Ferreira de Lima, Bruno Albert e Edmar Candeia.

Agradeço aos professores Joseana Fechine, Eustáquio Rangel, Tiago Massoni, Jacques Sauvé, Raquel Lopes e Livia Sampaio, dos quais fui aluna durante a pós-graduação ou que pude receber apoio na realização das mais diversas atividades.

Aos demais amigos, meu sincero agradecimento pelas mais diversas razões. Alguns deles listo aqui de forma breve: Adriana Carla, Amanda Oliveira, Catuxe Varjão, Cheyenne Isidro, Débora Prazeres, Débora Magalhães, Diego Cavalcanti, Expedito Lopes, Franklin Marquezino, Georgina Freitas, João Felipe Ouriques, Larissa Lucena, Lucelya Arruda, Mariana Franco, Mariana Romão, Natália Damião, Neves, Odilon Lima, Paula Pérez e toda a galera de Comp51. Um grande obrigada também aos colegas da Direção do Centro de Ciências e Tecnologia da UFCG que, apesar do pouco tempo de trabalho, sei que fiz amigos para toda a vida.

Muito obrigada a todos os meus familiares, dentre os quais aqueles que se fizeram mais presente nos últimos tempos: Vovó Zefinha, Tia Com, Tia Edinha, Tio Edson, Tia Graça, Tia Goretti e Tio Pedro, Tia Jazetti, Tia Beta e Tio Estanislao, Tio Jaime, Tia Sandra e Tio Dido, Raquel e Rafael Bertucci, Dimas, Nara, Ana Luísa, Ana Laura, Marianne, Washington, Luís Felipe e Luís Augusto.

Agradeço aos professores e funcionários da Copin: Prof. Hyggo Almeida, Vera Oliveira, Rebeka Lemos e Ana Lúcia Guimarães. Agradeço também aos professores e funcionários do Departamento de Sistemas e Computação.

Meus agradecimentos a Universidade Federal de Campina Grande, ao Centro de Engenharia Elétrica e Informática, ao Departamento de Sistemas e Computação e ao Instituto de Estudos em Computação e Informação Quânticas (IQuanta) que apoiaram integralmente a realização do meu trabalho. Agradeço também ao Governo Brasileiro, por meio da CAPES, pelo apoio financeiro fornecido para execução das atividades deste mestrado.

Epígrafe

Plus animi est inferenti periculum, quam propulsanti.

(*Há mais espírito no ataque do que na defesa*)

Titus Livius, 59 a.C. – 17 d.C.

Dedicatória

Àqueles que sempre me inspiraram.

Conteúdo

Lista de Acrônimos

BM	Blum-Micali
CSPRNG	Gerador de Números Pseudo-Aleatórios Criptograficamente Seguro
MT	Máquina de Turing
PRNG	Gerador de Números Pseudo-Aleatórios
QFT	Transformada Quântica de Fourier
RNG	Gerador de Números Aleatórios

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos e Códigos-Fonte

Capítulo 1

Introdução

Muitos métodos estatísticos utilizados pela Engenharia, Ciência da Computação e Ciências Naturais demandam a utilização de números aleatórios para simulação de processos físicos e biológicos, execução de algoritmos, dentre outros. Estes números são obtidos através de geradores de números aleatórios (RNGs – *Random Numbers Generators*), que fazem uso de determinados processos físicos, tais como decaimento de uma substância química ou turbulência em discos rígidos, para geração de tais seqüências [van Tilborg 2005].

Sabe-se que computadores digitais não são capazes de gerar tais números e que nem sempre é conveniente ou viável conectar uma *hardware* externo para obtenção de números aleatórios. Uma maneira de contornar esta limitação é por meio da utilização de geradores de números pseudo-aleatórios (PRNGs – *Pseudorandom Numbers Generators*). Este tipo de gerador é implementado por algoritmos, os quais são inicializados com alguns parâmetros, e produzem seqüências que se assemelham àquelas advindas de RNGs. Diversos tipos de PRNGs são definidos na literatura, a citar: geradores congruentes lineares, geradores congruentes não-lineares, geradores de deslocamento retro-alimentados, geradores baseados em autômatos celulares, geradores baseados em sistemas caóticos, dentre diversos outros [Gentle 2003].

A escolha de um gerador não pode ser feita ao acaso, deve levar em consideração os requisitos da aplicação a que se destina. Uma maneira de mensurar a qualidade de um PRNG é por meio da execução de uma bateria de testes estatísticos [Marsaglia 1995; Rukhin et al. 2008]. Porém, quando se trata de geradores para fins criptográficos, a avaliação por testes estatísticos não é suficientemente adequada para assegurar a escolha de um bom gerador.

Em esquemas de Criptografia, os geradores são elementos primordiais e, portanto, considerados *primitivas criptográficas*. Neste domínio, a utilização de seqüências pseudo-aleatórias está intimamente ligada à manutenção do segredo de uma mensagem a ser transmitida, o que implica em requisitos mais restringentes na escolha do gerador a ser utilizado. Estes requisitos adicionais são necessários porque as baterias de testes não compreendem todos os algoritmos de tempo polinomial capazes de prever padrões ou descrever a saída do gerador. Esta limitação das baterias de testes pode esconder vulnerabilidades do gerador e isto, num âmbito criptográfico, pode favorecer uma quebra de segurança [Paar and Pelzl 2010].

Para este tipo de aplicação, sugere-se o uso de geradores de números pseudo-aleatórios criptograficamente seguros (CSPRNGs– *Cryptographically Secure Pseudorandom Numbers Generators*), um tipo especial de PRNG [Blum and Micali 1984]. Este tipo de gerador é comumente baseado em problemas da Teoria dos Números para os quais não se conhece soluções eficientes, ou seja, baseiam-se em hipóteses de intratabilidade [Sidorenko 2008; Goldreich 2005].

Uma vez que são componentes de diversos esquemas de Criptografia, os PRNGs também podem ser alvos de ataques. Estes ataques podem ser de diversos tipos e visam a descoberta da semente, a reprodução das seqüências produzidas em um dado intervalo de tempo, a obtenção de saídas passadas do gerador, etc [Kelsey et al. 1998]. Estas informações podem favorecer um adversário do esquema criptográfico, auxiliando-o na descoberta de uma mensagem secreta. No caso dos CSPRNGs, em particular, os ataques tornam-se mais difíceis, pois a complexidade dos ataques conhecidos atualmente é de ordem exponencial, considerando o paradigma da Computação Clássica. Porém, é preciso investigar a vulnerabilidade de tais geradores em relação a ataques formulados em outros paradigmas computacionais.

A Computação Quântica é um paradigma computacional baseado nos postulados da Mecânica Quântica e, portanto, demanda que a Física Quântica seja levada em conta nos passos da Computação. Este modelo de computação foi proposto por Deutsch [1985] com a definição de uma Máquina de Turing Quântica.

A proposição de algoritmos eficientes para problemas em que uma solução de tempo polinomial da Computação Clássica não era conhecida têm motivado o estudo e a proposição de novos algoritmos de acordo com o paradigma da Computação Quântica. Dentre os al-

goritmos já existentes, a busca quântica [Grover 1997] e o algoritmo de Shor [1997] são os principais exemplos de ganhos significativos em relação aos algoritmos clássicos equivalentes.

Neste contexto, identifica-se um problema:

> *A vulnerabilidade de geradores de números pseudo-aleatórios contra ameaças da Computação Quântica não é conhecida.*

Este problema aponta questões de pesquisa a serem investigadas: Será que é possível elaborar ataques quânticos à geradores de números pseudo-aleatórios, em particular aos CSPRNGs? Em caso afirmativo, será que há ganhos (i.e., diminuição na complexidade computacional) em relação aos ataques clássicos equivalentes? O trabalho de mestrado desenvolvido se propôs a esclarecer estas questões.

1.1 Objetivo do Trabalho

O objetivo geral deste trabalho é a proposição de ataques quânticos a geradores de números pseudo-aleatórios com o intuito de melhor caracterizar a vulnerabilidade dos mesmos contra ameaças da Computação Quântica. Para tanto, alguns objetivos específicos foram contemplados, a citar:

1. Caracterização adequada do problema tratado e identificação de contribuições e trabalhos relacionados;
2. Investigação da possibilidade de adaptação de algoritmos quânticos existentes para a definição de novos ataques;
3. Definição de algoritmos quânticos que implementem ataques a geradores de números pseudo-aleatórios;
4. Investigação da existência de ganhos dos algoritmos quânticos propostos em relação às suas contrapartidas clássicas.

A solução proposta caracteriza-se por um algoritmo quântico de ataque ao CSPRNG de Blum-Micali. Este ataque baseia-se no algoritmo quântico de busca [Grover 1997] e faz

uso do algoritmo quântico para o logaritmo discreto [Shor 1997] com o intuito de recuperar o estado interno do gerador e, assim, comprometer a imprevisibilidade do mesmo. Este ataque possui ganhos em relação ao ataque análogo formulado no paradigma da Computação Clássica.

Outros resultados também foram alcançados. O primeiro deles diz respeito a determinação do número de bits que devem ser interceptados para que um adversário realize o ataque com sucesso. Os demais resultados tratam da generalização deste ataque, tornando possível a sua utilização em ataques a outros geradores além do Blum-Micali e também sob diferentes condições da seqüência de bits interceptada.

1.2 Relevância

O trabalho realizado colabora para as áreas da Criptoanálise e da Computação Quântica. Dado que PRNGs e CSPRNGs são primitivas criptográficas, isto é, componentes essenciais de diversos sistemas criptográficos, desenvolver ataques e analisar a segurança de geradores pode trazer resultados para aplicações criptográficas amplamente utilizadas. Esta afirmação é apoiada pela literatura, a qual ressalta a necessidade de desenvolver trabalhos desta natureza.

Em um *Request for Comments* que aborda requisitos de aleatoriedade para segurança, Eastlake et al. [2005] afirmam que a utilização de geradores pseudo-aleatórios em algoritmos criptográficos pode significar pseudo-segurança. Um adversário pode realizar a tarefa de reproduzir as saídas do gerador e então obter acesso a um segredo de forma simplificada, por exemplo, por meio da redução drástica de um espaço de busca. Uma vez que tal documento reporta requisitos que as aplicações criptográficas reais devem seguir, não é possível desconsiderar a existência e a viabilidade de tais ataques. Os autores reforçam ainda que proposição de PRNGs e CSPRNGs atualmente é uma tarefa complexa.

Kelsey et al. [1998] apontam alguns aspectos que reforçam a preocupação com o uso e a possibilidade de ataques a PRNGs em sistemas de Criptografia. O primeiro deles é que PRNGs são pontos únicos de falha em muitos sistemas criptográficos. Isto significa que a ocorrência de um ataque bem-sucedido a este componente do sistema criptográfico pode comprometer toda a segurança do sistema. O segundo aspecto levantado pelos autores diz que a escolha inadequada ou má inicialização de um PRNG pode tornar insignificante a

escolha de bons algoritmos e protocolos de cifragem, uma vez que o bom funcionamento destes últimos é fortemente dependente da qualidade das saídas do gerador associado. Por fim, os autores argumentam que ainda não há um consenso sobre a extensão dos possíveis ataques a geradores de números pseudo-aleatórios, nem tampouco completo conhecimento das limitações em utilizar um determinado tipo de gerador.

O trabalho de Sidorenko e Schoenmakers [2005] analisa e ressalta a importância de investigar aspectos acerca da segurança e inicialização de CSPRNGs. Kang [2005] argumenta que não há muitos trabalhos teóricos sobre o assunto, motivando e ressaltando a importância dos mesmos.

Considerando os relatos da literatura apresentados, na área de Criptoanálise, o trabalho realizado: (i) melhora o conhecimento acerca dos possíveis ataques aos PRNGs e CSPRNGs utilizados em aplicações criptográficas do mundo real; (ii) caracteriza algoritmos de ataque a geradores; (iii) analisa a segurança deste componente de diversos sistemas criptográficos contra ameaças da Computação Quântica; e (iv) aponta novos requisitos de segurança para PRNGs e CSPRNGs.

No tocante à Computação Quântica, o trabalho também traz contribuições, a citar: (i) a proposição de algoritmos de acordo com este paradigma; (ii) a caracterização de novas ameaças a sistemas de criptografia clássicos; e (iii) a investigação de ganhos em relação aos algoritmos clássicos que implementam ataques a geradores.

Os resultados deste trabalho também contribuem para os objetivos do IQuanta – Instituto de Estudos em Computação e Informação Quânticas, no qual o trabalho foi desenvolvido.

1.3 Metodologia

A metodologia empregada neste trabalho foi composta de atividades de estudo e pesquisa apresentadas a seguir:

1. Realização de revisões bibliográficas;
   (a) Revisão bibliográfica sobre geradores aleatórios e pseudo-aleatórios;
   (b) Revisão bibliográfica sobre Computação Quântica e Algoritmos Quânticos;

2. Avaliação da viabilidade de adaptação dos algoritmos quânticos existentes para elaboração de ataques quânticos a geradores;
   (a) Avaliação do algoritmo quântico de busca;
   (b) Avaliação do algoritmo quântico para o logaritmo discreto;
3. Proposição de ataques criptoanalíticos com Computação Quântica;
   (a) Elaboração de ataque ao gerador Blum-Micali;
   (b) Elaboração de provas de corretude para o algoritmo proposto;
   (c) Análise de complexidade;
   (d) Verificação da existência de ganhos em relação ao paradigma clássico;
   (e) Determinação do número de bits requeridos para o ataque;
4. Generalização do ataque proposto;
   (a) Generalização para geradores com múltiplos predicados difíceis;
   (b) Generalização para a Construção de Blum-Micali;
   (c) Generalização para bits não consecutivos;
   (d) Generalização para menos bits que o requerido;
5. Escrita de artigos científicos;
6. Escrita e defesa da dissertação.

Os resultados alcançados nestas atividades estão documentados ao longo desta dissertação.

1.4 Estrutura da Dissertação

Este capítulo apresentou as informações gerais sobre o trabalho realizado, definindo o problema abordado, os objetivos do trabalho, a relevância e a metodologia utilizada.

O Capítulo 2 apresenta a fundamentação teórica sobre geradores aleatórios e pseudo-aleatórios. Este capítulo apresenta as diferenças entre geradores aleatórios, pseudo-aleatórios

e criptograficamente seguros. Há ainda a caracterização de uma taxonomia de ataques a geradores, que será utilizada ao longo desta dissertação.

A fundamentação teórica sobre os algoritmos utilizados na dissertação encontra-se no Capítulo 3. Este capítulo contempla os algoritmos quânticos de busca e o algoritmo quântico para o logaritmo discreto. Ambos são utilizados na elaboração da solução proposta neste trabalho. Caso o leitor não seja familiarizado com Computação Quântica, recomenda-se a prévia leitura dos Apêndices A e B, que apresentam os fundamentos da Computação Quântica e de Circuitos Quânticos, respectivamente.

O primeiro grupo de contribuições desta dissertação encontra-se no Capítulo 4, que contempla a apresentação do ataque quântico ao gerador Blum-Micali. A organização deste capítulo é descrita a seguir. A seção 4.1 apresenta os conceitos relativos a este gerador, seguido da elaboração de um ataque clássico ao mesmo, descrito na Seção 4.2. Algumas considerações sobre este ataque são delineadas na Seção 4.3. A partir deste ataque, desenvolve-se o ataque quântico ao gerador de Blum-Micali, cuja descrição é divida em duas partes: identificação do representante, descrita na Seção 4.4.1, e amplificação da amplitude, descrita na Seção 4.4.2. Para facilitar a compreensão deste algoritmo, um exemplo detalhado é apresentado na Seção 4.4.3. Por fim, uma análise detalhada do ataque proposto é realizada na Seção 4.5, que contempla a análise de complexidade, investigando ganhos em relação à contrapartida clássica; considerações sobre a impossibilidade de uso do algoritmo quântico do logaritmo discreto para recuperação do representante; e também aspectos sobre a construção das portas requeridas.

A segunda parte das contribuições deste trabalho encontra-se descrita no Capítulo 5. A partir do ataque descrito no capítulo anterior, generalizações são propostas, tornando-o apto a atacar um conjunto mais amplo de geradores além do Blum-Micali. Estas generalizações englobam a definição de portas, montagem de circuitos e outras adaptações que permitem que o ataque proposto funcione adequadamente contra geradores com múltiplos predicados (Seção 5.1), geradores da Construção de Blum-Micali (Seção 5.2), e em situações em que o adversário captura bits não consecutivos (Seção 5.3) ou então detém menos bits que o requerido (Seção 5.4). Embora esta descrição seja feita de forma distinta, é possível utilizar as generalizações propostas de forma ampla, por exemplo, para efetuar um ataque a um gerador da Construção de Blum-Micali em que o adversário recuperou menos bits que o

requerido.

Por fim, o Capítulo 6 apresenta as considerações finais desta dissertação. A primeira parte destas considerações, relatada na Seção 6.1, apresenta conclusões e considerações sobre as contribuições propostas, discutindo a relação das mesmas com outros trabalhos da literatura, apontando vantagens e limitações. Ainda nesta seção são pontuadas as publicações realizadas em decorrência dos resultados alcançados. A segunda parte das considerações, descrita na Seção 6.2, contempla sugestões de trabalhos futuros identificados ao longo do desenvolvimento desta dissertação.

O Apêndice C contempla dois artigos produzidos ao longo deste trabalho. O primeiro deles foi publicado nos anais do *International Telecommunications Symposium* 2010, e o segundo nos anais do III Workshop-Escola de Computação e Informação Quânticas.

É importante salientar que todas as atividades desta dissertação foram desenvolvidas no Instituto de Estudos em Computação e Informação Quânticas (IQuanta), sediado na Universidade Federal de Campina Grande. O IQuanta é uma instituição de direito privado, sem fins lucrativos, que promove o desenvolvimento de conhecimentos e de tecnologias nas áreas da Computação e da Informação Quânticas, incentivando parcerias, a troca de informações e a discussão de temas relacionados ao crescimento brasileiro nessas áreas, em todos os seus aspectos: pesquisa, desenvolvimento, ensino e serviço.

Capítulo 2

Geradores de Números Aleatórios e Pseudo-Aleatórios

Este capítulo apresenta os conceitos elementares relacionados aos geradores de números aleatórios e pseudo-aleatórios. Inicialmente, os geradores de números aleatórios são apresentados na Seção 2.1. Posteriormente, na Seção 2.2, são conceituados os geradores de números pseudo-aleatórios. Os geradores pseudo-aleatórios voltados para fins criptográficos, ou seja, os geradores de números pseudo-aleatórios criptograficamente seguros, são apresentados na Seção 2.3. Por fim, na Seção 2.4 são apresentados os ataques possíveis contra tais geradores.

Neste trabalho, os termos "gerador de bits", "gerador de seqüências" ou simplesmente o termo "gerador" são utilizados sem distinção, denotando um gerador de números qualquer.

2.1 Geradores de Números Aleatórios

Muitos métodos estatísticos utilizados pela Engenharia e Ciências Naturais demandam a utilização de *números aleatórios* para simulação de processos físicos e biológicos. A utilização destes números também se aplica a outros domínios, por exemplo, na Criptografia, para permitir a troca segura de dados, e nos sistemas operacionais, possibilitando a execução de determinados algoritmos [Gentle 2003].

Uma seqüência de números é dita ser *aleatória* se (i) os bits da representação binária dos números da seqüência assumirem os valores "falso" ou "verdadeiro" de modo *equiprovável* e se (ii) a obtenção de tais números for feita de forma independente estatisticamente [Rukhin

et al. 2008]. Uma definição alternativa, enuncia que seqüências aleatórias finitas de números não podem ser descritas por alguma subseqüência de si mesmas [L'Ecuyer 1990].

Para obtenção de números aleatórios utilizam-se *geradores de números aleatórios* (RNGs – *Random Numbers Generators*) que, para produzirem seqüências de números, possuem uma fonte não-determinística (fonte de entropia) em conjunto com alguma função de processamento (processo de destilação da entropia), necessária para minimizar a geração de subseqüências não-aleatórias decorrentes da fonte de entropia.

Vários tipos diferentes de fontes de entropia são utilizadas em RNGs, a citar: decaimento radioativo de uma substância química [Gentle 2003], turbulência de ar em discos rígidos [Davis et al. 1994], freqüências capturadas por microfones em ambientes ruidosos [van Tilborg 2005], dentre outros. Apesar de gerarem números aleatórios, na prática alguns RNGs demandam tempo para geração de seqüências e os resultados de algumas fontes de entropia podem ser previsíveis (fontes de baixa entropia), o que torna inviável a utilização deste geradores sem outras técnicas ou tecnologias associadas [Rukhin et al. 2008].

Embora produzam seqüências aleatórias, nem sempre RNGs estão disponíveis nos sistemas computacionais. Para contornar esta limitação, utiliza-se uma alternativa algorítmica: os geradores de números pseudo-aleatórios, discutidos na seção a seguir.

2.2 Geradores de Números Pseudo-Aleatórios

Computadores digitais não são capazes de gerarem números aleatórios e, em algumas situações, não é conveniente conectar uma *hardware* externo para obtenção de tais números. Em alguns casos, sem prejuízo ao resultado final da aplicação, é possível recorrer à utilização de seqüências numéricas advindas de *geradores de números pseudo-aleatórios* (PRNGs – *Pseudorandom Numbers Generators*).

Um PRNG é um algoritmo determinístico que, dado um valor aleatório inicial, é capaz de produzir uma seqüência de números que parece ser aleatória. Matematicamente, estes geradores podem ser representados por uma função determinística recursiva f que toma os k números gerados anteriormente para determinação de um novo número:

$$x_i = f(x_{i-1}, \ldots, x_{i-k}) \tag{2.1}$$

em que $x_{i-1}, \ldots, x_{i-k}$ são números inteiros.

A quantidade k de números anteriores utilizados denomina-se a *ordem* do gerador e o conjunto de valores que é fornecido inicialmente denomina-se *semente*. Uma vez que a quantidade de números fornecida é limitada, a seqüência resultante irá repetir. O comprimento da seqüência de números gerada até a primeira repetição denomina-se *período* do gerador. A literatura apresenta uma vasta gama de definições de PRNGs, a citar: geradores congruentes lineares, geradores congruentes não-lineares, geradores de deslocamento retroalimentados, geradores baseados em autômatos celulares, geradores baseados em sistemas caóticos, dentre outros [Gentle 2003].

O *estado interno do gerador no tempo* i, denotado por $X(i)$, é um conjunto ordenado contendo a semente e os valores produzidos pela recursão da função f até o instante i, i.e., $X(i) = \{x_i, x_{i-1}, \ldots, x_1, x_0\}$. O valor $x_i \in X(i)$ é denominado *representante* do estado interno do gerador no tempo i.

Existem diversas aplicações em que os PRNGs são utilizados em substituição aos RNGs. Por exemplo, em sistemas operacionais, PRNGs possibilitam a execução de algoritmos de escalonamento e auxiliam na identificação e recuperação de *deadlocks* [Woodhull and Tanenbaum 2008]. Porém, diante dos diversos tipos de PRNGs, a escolha do gerador adequado para uma determinada aplicação deve levar em consideração aspectos estatísticos, que podem ser avaliados por uma bateria de testes (ver Guedes [2009], Rukhin et al. [2008] e Marsaglia [1995] para uma completa descrição destes testes).

2.3 Geradores de Números Pseudo-Aleatórios Criptograficamente Seguros

Embora possam ser aplicados em diversos domínios, nem todo PRNG pode ser utilizado em aplicações de Criptografia. Isto acontece porque as baterias de testes estatísticos, amplamente utilizadas para avaliar a qualidade de PRNGs, não são suficientes para detectar todos os possíveis padrões na saída destes geradores [Paar and Pelzl 2010]. Para exemplificar esta afirmação, o caso dos geradores congruentes lineares é apresentado.

Os geradores congruentes lineares foram propostos por Lehmer [1951]. Neste tipo de gerador, um único número, a partir da semente, determina o seu sucessor por meio de uma

multiplicação seguida de uma redução modular, como mostrado a seguir:

$$x_i \equiv a \cdot (x_{i-1} + c) \bmod m \tag{2.2}$$

em que a é um multiplicador, c é um incremento e m é o módulo do gerador. Há casos em que geradores congruentes lineares inicializados com determinadas combinações de x_0, a, c e m passam em diversos testes estatísticos. Mas, quando as seqüências produzidas são dispostas em reticulados, revelam padrões não observados antes e que permitem que as produções do gerador possam ser previstas [Marsaglia 1968], como mostrado na Figura 2.1. No contexto da Criptografia, por exemplo, tal previsibilidade poderia comprometer toda a segurança de um sistema criptográfico.

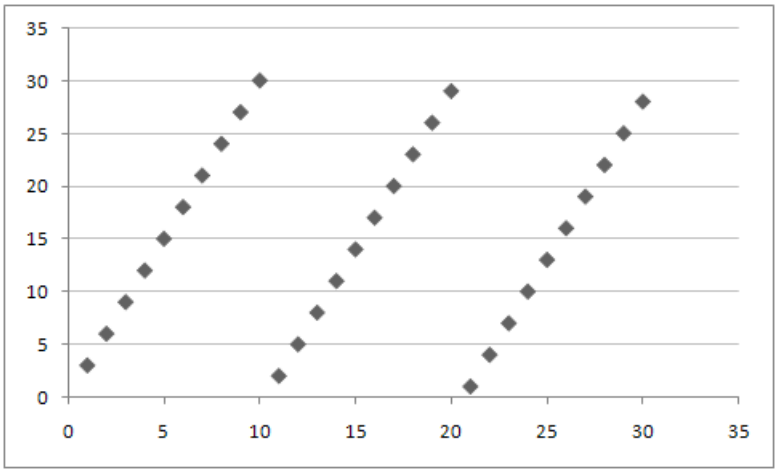

Figura 2.1: Reticulado em $\mathbb{R}^2$ obtido a partir do gerador congruente linear simples $x_i \equiv (7 \cdot x_{i-1}) \bmod 31$ alimentado com semente igual a 27.

Há uma forte relação entre Criptografia e números pseudo-aleatórios, pois a segurança de muitos algoritmos criptográficos depende da escolha aleatória de chaves e seqüências de bits [Delfs and Knebl 2007]. Diante disto, os requisitos para os geradores pseudo-aleatórios utilizados neste domínio tornam-se mais restringentes e, em decorrência, nem todos os PRNGs podem ser utilizados.

Um gerador pseudo-aleatório é dito ser criptograficamente seguro (CSPRNG – *Cryptographically secure pseudorandom numbers generator*) se as seqüências produzidas por este gerador não puderem ser distinguidas de seqüências aleatórias pelo julgamento efetuado por um algoritmo de tempo polinomial [Blum and Micali 1984; Yao 1982].

Para que haja dificuldade em distinguir as seqüências produzidas pelos CSPRNGs, é necessário que estes se baseiem em problemas difíceis computacionalmente[1]. Levando isto em consideração, a construção dos CSPRNGs é feita a partir do uso de funções *one-way*.

Funções *one-way* são todas as funções fáceis de computar, mas difíceis de inverter. Isto é, dada uma função f, em que $f : \{0,1\}^* \rightarrow \{0,1\}^*$, diz-se que f é uma função *one-way* se: dada uma entrada x, há um algoritmo eficiente que obtém $f(x)$, mas que dado $f(x)$, não há algoritmo eficiente para obter a pré-imagem x sob a função f.

A prova de que funções *one-way* são difíceis computacionalmente ainda é uma questão em aberto, pois decorre da questão $\mathcal{P}$ vs $\mathcal{NP}$. Em virtude disto, diz-se que tais funções se baseiam em hipóteses de intratabilidade, já que esta última não pode ser provada. A maioria das funções *one-way* são construídas a partir de problemas da Teoria dos Números, a exemplo da fatoração, logaritmo discreto, dentre outros [Goldreich 2005].

Considerando uma função *one-way* f e o valor $f(x)$, é possível obter, em tempo polinomial, certas informações a respeito de x. Isto é feito com o uso de predicados, ou seja, funções que, dado $f(x)$, resultam em valores binários que revelam alguma informação sobre x. No escopo de CSPRNGs, há o interesse em predicados difíceis (*hard-core predicates*), ou seja, aqueles que fornecem um bit, mas que ainda mantém a intratabilidade de obter x.

Com isto, os CSPRNGs são construídos com uma função ou mais funções *one-way* e predicados difíceis. A semente é um valor inicial que alimenta a recursão da(s) função(ões) *one-way* e as seqüências de bits produzidas são resultado dos predicados difíceis paras as funções *one-way* associadas [Håstad et al. 1999].

Dada a definição de CSPRNGs apresentada, uma característica resultante destes geradores é que são *imprevisíveis*, ou seja, dada uma seqüência de n bits consecutivos, não há ou não se conhece algoritmo de tempo polinomial capaz de predizer o próximo bit com mais de 50% de chance de acerto [Paar and Pelzl 2010].

No domínio Criptográfico, a utilização de seqüências pseudo-aleatórias é de extrema importância, pois as mesmas são necessárias em esquemas de autenticação recíprocos, geração de chaves de sessão, geração de chaves para o algoritmo RSA, esquemas de assinatura digital, dentre outros [Hoffstein et al. 2008]. Afirma-se até que a utilização de CSPRNGs é

[1]Problemas difíceis computacionalmente são aqueles para os quais não se conhece ou não existe uma solução de tempo polinomial [Toscani and Veloso 2002].

um requisito de toda aplicação criptográfica bem projetada [Kelsey et al. 1998]. Ainda neste domínio, diante dos fortes requisitos matemáticos que devem ser respeitados, a proposição de novos CSPRNGs é considerada uma tarefa complexa [Eastlake et al. 2005]

2.4 Ataques a Geradores de Números Pseudo-Aleatórios

Entende-se por *Criptoanálise* a subárea da Criptologia responsável pelo estudo e desenvolvimento de meios que promovam a descoberta de informações secretas, em particular, aquelas ligadas aos algoritmos de encriptação. A Criptoanálise pode ser divida em ataques analíticos, que exploram a estrutura interna do elemento sob ataque, e em ataques de força-bruta, que tratam o elemento sob ataque como uma caixa-preta e testam todos valores possíveis até descobrir o dado secreto [van Tilborg 2005]. Ataques criptoanalíticos também podem ser aplicados a geradores de números pseudo-aleatórios. Tais ataques podem visar a descoberta da semente, a reprodução das seqüências produzidas em um dado intervalo de tempo, a obtenção de saídas passadas do gerador, etc.

Uma vez que os PRNGs e CSPRNGs são componentes de diversos sistemas computacionais, inclusive os de fins criptográficos, é essencial analisar a vulnerabilidade dos mesmos contra ataques criptoanalíticos. Se há um ataque eficiente para um gerador utilizado em uma dada aplicação criptográfica, é bem possível que a segurança da aplicação possa estar comprometida diante da vulnerabilidade do gerador, não importando a segurança dos demais algoritmos criptográficos utilizados.

Para classificar os ataques possíveis a um gerador de números pseudo-aleatórios, neste trabalho será utilizada como referência a taxonomia de Kelsey [Kelsey et al. 1998], apresentada detalhadamente na próxima seção. Além disso, em se tratando de ataques à geradores, assume-se que Engenharia Social não é utilizada para obter qualquer tipo de informação mantida em segredo.

2.4.1 Taxonomia de Kelsey

A classificação de ataques a geradores de números pseudo-aleatórios ainda não é consenso na literatura. Alguns autores afirmam que apenas os ataques de distinção são de interesse, ou seja, aqueles que permitem avaliar quando é possível ou não distinguir as seqüências pro-

duzidas pelo gerador de seqüências verdadeiramente aleatórias [Goldreich 2005]. Sidorenko e Schoenmakers [2005], por exemplo, afirmam que além dos ataques de distinção, existem os ataques de recuperação de estado, isto é, aqueles que recuperam alguma informação que permite que um adversário reproduza as seqüências do gerador.

Neste trabalho, será considerada como referência a *taxonomia de Kelsey* [Kelsey et al. 1998]. A opção por seguir a classificação proposta por estes autores se deve ao fato desta prever mais possibilidades de ataques e de suas conseqüências. Além disto, esta taxonomia é fortemente baseada em ataques praticáveis contra geradores utilizados em sistemas criptográficos, ou seja, mais fiel aos ataques do mundo real.

Assume-se que o termo *adversário* denota um indivíduo ou algoritmo interessado em quebrar a aparente imprevisibilidade de um gerador. Levando isto em consideração e também utilizando a notação apresentada na Seção 2.2, de acordo com a taxonomia de Kelsey, ataques contra geradores de números pseudo-aleatórios podem ser classificados como:

1. **Ataques Criptoanalíticos Diretos**. Ocorrem quando o adversário é capaz de distinguir entre as saídas do gerador de números pseudo-aleatórios e saídas aleatórias. Este tipo de ataque, também conhecido como ataque de distinção, foi proposto inicialmente por Yao [1982];

2. **Ataques Baseados em Entradas**. Acontecem quando o adversário é capaz de usar algum conhecimento ou controle das entradas do PRNG para auxiliar na criptoanálise. Três subcategorias desse ataque são definidas em função do tipo de entrada utilizada: ataques baseados em entradas escolhidas, repetidas ou conhecidas;

3. **Ataques de Comprometimento de Estado**. Neste tipo de ataque, o adversário utiliza-se das vantagens de um ataque anterior para obter informações sobre o estado interno do gerador. Uma seqüência de bits produzida pelo PRNG é um exemplo de elemento de interesse a ser obtido pelo ataque prévio do adversário. Existem quatro classificações para ataques de comprometimento de estado:

   (a) **Ataque de Retorno**. Também conhecido como *backtracking attack*. Neste tipo de ataque o adversário conhece $x_i \in X(i)$ e utiliza esta informação para obter os outros elementos do estado interno do gerador, i.e., $x_k \in X(i)$ em que $k < i$;

(b) **Ataque de Comprometimento Permanente**. Acontece quando o adversário recupera $x_i \in X(i)$ e com esta informação é capaz de conhecer estados anteriores do gerador ($x_k \in X(i)$, em que $k < i$) e também prever estados futuros ($x_j \in X(i+\epsilon)$, em que $j > i$, $\epsilon > 0$ e $j \leq i + \epsilon$). Este tipo de ataque compromete totalmente a imprevisibilidade do gerador;

(c) **Ataque de Descoberta Iterativa**. Utiliza conhecimento de $x_i \in X(i)$ e de intervenções na saída do gerador para determinar $x_{i+\epsilon} \in X(i+\epsilon)$, em que $\epsilon > 0$. Neste tipo de ataque não é necessário que o adversário saiba as saídas do gerador produzidas no intervalo Δ_ϵ, mas é requerido que seja capaz de prevê-las;

(d) **Ataque *Meet-in-the-Middle***. Consiste na combinação de um ataque de descoberta iterativa com um ataque de retorno. Dado o conhecimento de $x_i \in X(i)$ e de $x_{i+2\cdot\epsilon} \in X(i + 2 \cdot \epsilon)$, este tipo de ataque permite que o adversário recupere o estado $x_{i+\epsilon} \in X(i+\epsilon)$.

É importante salientar que nem todo PRNG ou CSPRNG é susceptível a todos os tipos de ataques apresentados. De acordo com os autores, o ataque criptoanalítico direto, por exemplo, só se aplica a PRNGs utilizados em esquemas criptográficos cuja saída pode ser obtida diretamente. Isto, por exemplo, implicaria na inviabilidade deste ataque contra um gerador utilizado no Triple-DES [Kelsey et al. 1998].

Notas do Capítulo

Este capítulo contempla a fundamentação teórica sobre geradores de números aleatórios e pseudo-aleatórios. Foi visto que RNGs utilizam determinados processos físicos para a produção de seqüências de números aleatórios. Por outro lado, computadores digitais não são capazes de gerar tais números e, por isso, utilizam uma alternativa algorítmica: os geradores de números pseudo-aleatórios, que podem ter diferentes definições. Os PRNGs utilizados em sistemas de Criptografia são criptograficamente seguros, ou seja, são construídos a partir de funções *one-way* associadas a predicados difíceis e passam por baterias de teste de tempo polinomial. Uma vez que estes geradores são utilizados em aplicações criptográficas, ataques são praticáveis contra os mesmos. Quando bem sucedidos, estes ataques podem

comprometer a segurança de toda a aplicação criptográfica, não importando a qualidade dos algoritmos de encriptação escolhidos. Os ataques possíveis contra um gerador são diversos e, para referência, será utilizada a taxonomia de Kelsey.

Capítulo 3

Algoritmos Quânticos de Busca e do Logaritmo Discreto

Este capítulo apresenta a fundamentação teórica dos algoritmos quânticos utilizados na realização deste trabalho. Nesta apresentação será utilizada a notação de Dirac [1982] e será levado em consideração que o leitor é familiar com os conceitos da Computação e Circutos Quânticos. Para aqueles leitores não familiarizados com tais conceitos, sugere-se a leitura do Apêndice A, que apresenta os fundamentos da Computação Quântica, e do Apêndice B, que apresenta a notação dos circuitos quânticos.

Esta fundamentação contempla o algoritmo quântico de busca, proposto por Grover [1997], capaz de realizar busca de elementos em uma base de dados desordenada, e o algoritmo quântico para o problema do logaritmo discreto [Shor 1997]. Em ambos os casos, tais algoritmos serão apresentados em função dos circuitos quânticos que os implementam.

3.1 Algoritmo Quântico de Busca

Uma das tarefas computacionais mais requisitadas é a *busca* de elementos, que consiste em verificar a presença (ou ausência) de um determinado item em uma base de dados *desordenada*. A busca é necessária para recuperar informações de clientes do bancos de dados de uma administradora de cartões de crédito, encontrar o endereço de um determinado site a partir de palavras-chave fornecidas a um buscador na internet, recuperar determinadas informações genéticas em um exame de DNA, dentre diversas outras.

A busca de um item i_0 em uma lista com N elementos pode ser generalizada por uma função $f : \{0, \ldots, N-1\} \rightarrow \{0, 1\}$, e o problema da busca consiste em encontrar $x \in \{0, \ldots, N-1\}$ tal que:

$$f(x) = \begin{cases} 1, & \text{caso } x = i_0 \\ 0, & \text{caso contrário} \end{cases} \tag{3.1}$$

Na computação clássica, para realizar a busca de um elemento, é necessário varrer o conjunto de todos os ítens até que (i) o elemento desejado seja encontrado, ou que (ii) seja possível concluir que este elemento não se encontra na base. Esta busca, no pior caso, varre todo os N elementos da base de dados e, no caso médio, varre $N/2$ elementos, ou seja, tem complexidade $O(N)$ e, portanto, um custo linear. Neste problema, assume-se que não há qualquer estrutura que evite a observação item a item.

Em 1996, Grover [1997] propôs um algoritmo quântico de busca, também conhecido como Algoritmo de Grover. Fazendo uso dos postulados da Mecânica Quântica, este algoritmo consegue realizar a busca de um elemento em uma lista desordenada com custo $O(\sqrt{N})$, ou seja, com ganho quadrático em relação à sua contrapartida clássica.

Nas seções a seguir, serão apresentados os elementos que compõem o algoritmo de Grover. Esta explanação foi construída a partir de informações adquiridas em diversos trabalhos [Grover 1997; Portugal et al. 2004; Nielsen and Chuang 2005; Hirvensalo 2001; Imre and Balazs 2005; Mermin 2007; Kaye et al. 2007], os quais devem ser consultados pelo leitor a fim de expandir seus conhecimentos.

A apresentação do algoritmo de busca quântica será feita como segue. Inicialmente, são apresentadas as notações e convenções que serão adotadas ao longo do capítulo. Posteriormente, são descritos os passos que compõem o algoritmo, por meio do circuito quântico que o resolve. Há uma interpretação geométrica, seguida da apresentação de uma extensão deste algoritmo que o torna apto a lidar com múltiplas soluções. Por fim, algumas considerações sobre este algoritmo são expostas.

3.1.1 Notação e Convenções

Na busca quântica, não se referenciam diretamente os elementos do espaço de busca, mas sim os *índices* destes elementos, que encontram-se no intervalo 0 até $N-1$ e são denotados

pela letra i. O elemento que se deseja encontrar é um elemento $i_0 \in [0, N-1]$, que satisfaz uma dada condição, diga-se $f(i_0) = 0$.

Por conveniência, assume-se que o tamanho da lista de dados é $N = 2^n$, em que $n \in \mathbb{N}$. Isto permite que qualquer índice possa ser representado por n bits.

Assume-se que há um oráculo capaz de reconhecer a solução do problema de busca em uma unidade de tempo. Além disso, supõe-se que não há qualquer ordenação na base de dados que possa auxiliar na resolução do problema.

Para fins de simplificação, assim como feito originalmente por Grover [1997], esta apresentação do algoritmo quântico de busca considera que na base de dados só há uma única solução para o problema. O caso de múltiplas soluções será explorado posteriormente, na Seção 3.1.4.

3.1.2 Apresentação do Algoritmo Quântico de Busca

A idéia do algoritmo quântico de busca consiste em *marcar* o elemento que está sendo procurado, caso ele exista. Posteriormente a esta marcação, há uma *amplificação da amplitude* do elemento que foi marcado. Em decorrência desta amplificação, há um aumento na probabilidade do elemento procurado ser encontrado e, simultaneamente, há a diminuição nas probabilidades dos demais elementos (não-soluções) serem encontrados.

O algoritmo quântico de busca faz uso de um registrador de n qubits para representar a entrada e outro registrador com apenas um qubit, utilizado para auxiliar a marcação do elemento procurado. O circuito que implementa este algoritmo encontra-se ilustrado na Figura 3.1, na qual o registrador que representa a entrada é evidenciado.

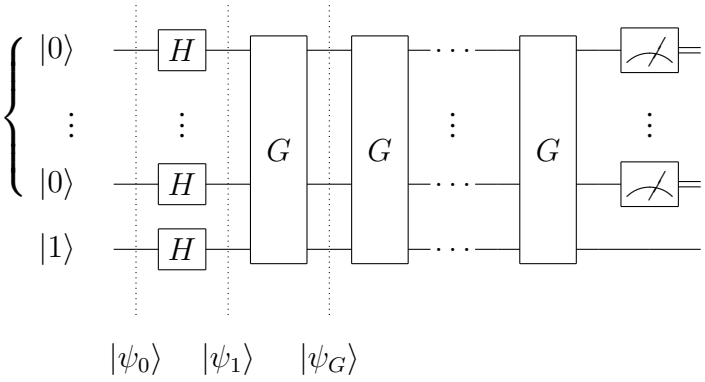

Figura 3.1: Circuito quântico que implementa o algoritmo quântico de busca.

Para a execução deste algoritmo, deve-se iniciar com $|0\rangle$ nos qubits referentes ao primeiro registrador, correspondendo aos elementos da base de dados onde será feita a busca (espaço de busca); e $|1\rangle$ no segundo registrador, correspondendo a um qubit auxiliar. Logo, o estado inicial $|\psi_0\rangle$ é dado pela seguinte expressão:

$$|\psi_0\rangle = |0\rangle^{\otimes n} |1\rangle \tag{3.2}$$

Após a inicialização, aplica-se o operador de Hadamard, com o intuito de criar uma superposição igualmente distribuída dos estados de entrada. Assim, o próximo estado que o sistema assume é dado por:

$$\begin{aligned} |\psi_1\rangle &= H^{\otimes(n+1)} |\psi_0\rangle && (3.3)\\ &= H^{\otimes n} |0\rangle^{\otimes n} \otimes H |1\rangle && (3.4)\\ &= \left(\frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle\right) \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) && (3.5)\\ &= \left(\frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle\right) |-\rangle && (3.6) \end{aligned}$$

Ao criar a superposição é possível, de fato, iniciar a busca quântica. Esta busca consiste na repetição de um conjunto de passos, conhecidos como *iteração de Grover*, implementados por um operador de Grover, denotado pela porta G.

O operador de Grover encapsula a atuação de duas portas, sendo a primeiro delas responsável pela inversão de fase; e a segunda pela amplificação da amplitude. A decomposição deste operador nas portas mencionadas encontra-se na Figura 3.2 e as subseções a seguir caracterizam e descrevem a atuação dos mesmos.

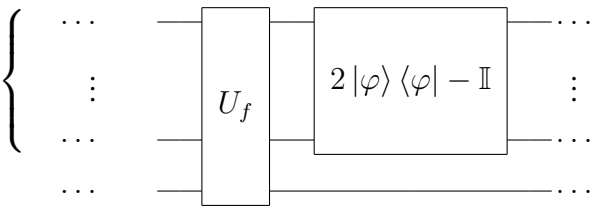

Figura 3.2: Decomposição de uma iteração de Grover.

Inversão da Fase

Suponha que exista um oráculo quântico, ou seja, uma caixa preta com a habilidade de *reconhecer* soluções para o problema de busca. No algoritmo de Grover, este oráculo é representado por um operador unitário U_f, que implementa a função f da Eq. (3.1). A atuação do operador U_f pode ser sintetizada da seguinte forma:

$$U_f \left|i\right\rangle \left|a\right\rangle \rightarrow \left|i\right\rangle \left|a \oplus f(i)\right\rangle \tag{3.7}$$

em que $\left|i\right\rangle$ é o estado do primeiro registrador, contendo n qubits; $\left|a\right\rangle$ é o segundo registrador, representado o qubit alvo; e o símbolo $\oplus$ denota a operação soma módulo 2.

A atuação de U_f pode ser exemplificada da seguinte forma, considerando $a = 0$ e $a = 1$:

$$U_f \left|i\right\rangle \left|0\right\rangle = \begin{cases} \left|i\right\rangle \left|1\right\rangle, & \text{se } i = i_0 \\ \left|i\right\rangle \left|0\right\rangle, & \text{se } i \neq i_0 \end{cases} \tag{3.8}$$

$$U_f \left|i\right\rangle \left|1\right\rangle = \begin{cases} \left|i\right\rangle \left|0\right\rangle, & \text{se } i = i_0 \\ \left|i\right\rangle \left|1\right\rangle, & \text{se } i \neq i_0 \end{cases} \tag{3.9}$$

$$\tag{3.10}$$

Na aplicação do operador U_f aos estados $\left|i\right\rangle \left|0\right\rangle$ e $\left|i\right\rangle \left|1\right\rangle$, é possível visualizar que há alteração no estado do segundo registrador quando o primeiro registrador contém o elemento procurado (i_0). Esta característica revela que a solução do problema de busca quântica é codificada no qubit alvo quando este não se encontra em uma superposição.

Considera-se que $\left|-\right\rangle = \frac{\left|0\right\rangle - \left|1\right\rangle}{\sqrt{2}}$. Ao retomar o estado $\left|\psi_1\right\rangle$ e efetuar a aplicação do oráculo U_f neste estado, têm-se:

$$
\begin{aligned}
|\psi_2\rangle &= U_f |\psi_1\rangle && (3.11)\\
&= U_f \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle |-\rangle && (3.12)\\
&= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} U_f |i\rangle |-\rangle && (3.13)\\
&= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} U_f |i\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) && (3.14)\\
&= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} \frac{1}{\sqrt{2}} U_f \left(|i\rangle |0\rangle - |i\rangle |1\rangle\right) && (3.15)\\
&= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} \frac{1}{\sqrt{2}} \left(|i\rangle |f(i)\rangle - |i\rangle |1 \oplus f(i)\rangle\right) && (3.16)\\
&= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} \frac{1}{\sqrt{2}} |i\rangle \left(|f(i)\rangle - |1 \oplus f(i)\rangle\right) && (3.17)
\end{aligned}
$$

De acordo com a definição de U_f, tem-se que:

$$
|i\rangle \left(|f(i)\rangle - |1 \oplus f(i)\rangle\right) = \begin{cases} |i\rangle \left(|1\rangle - |0\rangle\right), & \text{caso } i = i_0 \\ |i\rangle \left(|0\rangle - |1\rangle\right), & \text{caso } i \neq i_0 \end{cases} \tag{3.18}
$$

Substituindo esta definição na expressão anterior, é possível re-escrever $|\psi_2\rangle$:

$$
\begin{aligned}
|\psi_2\rangle &= \frac{1}{\sqrt{2^n}} \left\{ \left[\sum_{i=0, i\neq i_0}^{2^n-1} \frac{1}{\sqrt{2}} |i\rangle \left(|0\rangle - |1\rangle\right) \right] + \left[\frac{1}{\sqrt{2}} |i_0\rangle \left(|1\rangle - |0\rangle\right) \right] \right\} && (3.19)\\
&= \frac{1}{\sqrt{2^n}} \left[\left(\sum_{i=0, i\neq i_0}^{2^n-1} |i\rangle |-\rangle \right) - |i_0\rangle |-\rangle \right] && (3.20)\\
&= \left\{ \frac{1}{\sqrt{2^n}} \left[\sum_{i=0, i\neq i_0}^{2^n-1} |i\rangle |-\rangle + |i_0\rangle |-\rangle \right] \right\} - \frac{1}{\sqrt{2^n}} |i_0\rangle |-\rangle - \frac{1}{\sqrt{2^n}} |i_0\rangle |-\rangle && (3.21)\\
&= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle |-\rangle - \frac{2}{\sqrt{2^n}} |i_0\rangle |-\rangle && (3.22)\\
&= \left(\frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle - \frac{2}{\sqrt{2^n}} |i_0\rangle \right) |-\rangle && (3.23)
\end{aligned}
$$

Considerando $|\varphi\rangle$ como o seguinte estado:

$$|\varphi\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle \tag{3.24}$$

É possível denotar o estado $|\psi_2\rangle$ como segue:

$$|\psi_2\rangle = \left(|\varphi\rangle - \frac{2}{\sqrt{2^n}} |i_0\rangle \right) |-\rangle \tag{3.25}$$

Seja o estado $|\psi_{2a}\rangle$ equivalente ao estado $|\psi_2\rangle$, porém com uma notação explícita:

$$|\psi_{2a}\rangle = \frac{1}{\sqrt{2^n}} \cdot \{|0\rangle + |1\rangle + \ldots - |i_0\rangle + \ldots + |2^n - 2\rangle + |2^n - 1\rangle\} \otimes |-\rangle \tag{3.26}$$

Algumas considerações devem ser evidenciadas em relação a aplicação de U_f:

1. O módulo das amplitudes atribuídas aos elementos do primeiro registrador se mantém;
2. Houve uma "identificação" do elemento procurado por meio de uma inversão da fase, alterada para -1;
3. Embora o elemento desejado tenha sido identificado, esta informação só está acessível no nível quântico, pois uma medição retorna o valor $\left|-\frac{1}{\sqrt{2^n}}\right|^2 = \frac{1}{2^n}$, que é igual à probabilidade dos demais elementos da lista;
4. A porta U_f atua de forma simultânea em todos os elementos em superposição, graças ao *paralelismo quântico*; e
5. Não há qualquer alteração no segundo registrador, que se mantém no estado $|-\rangle$;

Com a aplicação da porta U_f, o elemento procurado encontra-se com a fase invertida. Porém, esta é uma informação que só está acessível no nível quântico. O próximo passo do algoritmo de Grover consiste em amplificar a amplitude deste elemento, possibilitado que a informação de que ele foi encontrado esteja acessível no nível clássico. Esta amplificação é detalhada a seguir.

Amplificação da Amplitude

O próximo passo do algoritmo de Grover consiste na aplicação do operador $(2\left|\varphi\right\rangle\left\langle\varphi\right| - \mathbb{I}) \otimes \mathbb{I}$, em que $\mathbb{I}$ denota a matriz identidade, ao estado $\left|\psi_2\right\rangle$, resultando em:

$$\begin{aligned}
\left|\psi_3\right\rangle &= \left[(2\left|\varphi\right\rangle\left\langle\varphi\right| - \mathbb{I}) \otimes \mathbb{I}\right]\left|\psi_2\right\rangle && (3.27)\\
&= \left[(2\left|\varphi\right\rangle\left\langle\varphi\right| - \mathbb{I}) \otimes \mathbb{I}\right]\left(\left|\varphi\right\rangle - \frac{2}{\sqrt{2^n}}\left|i_0\right\rangle\right)\left|-\right\rangle && (3.28)\\
&= \left[(2\left|\varphi\right\rangle\left\langle\varphi\right| - \mathbb{I})\left(\left|\varphi\right\rangle - \frac{2}{\sqrt{2^n}}\left|i_0\right\rangle\right)\right]\left|-\right\rangle && (3.29)\\
&= \left(2\left|\varphi\right\rangle\left\langle\varphi\right|\varphi\right\rangle - \left|\varphi\right\rangle - 2\left|\varphi\right\rangle\left\langle\varphi\right|\frac{2}{\sqrt{2^n}}\left|i_0\right\rangle + \frac{2}{\sqrt{N}}\left|i_0\right\rangle\Big)\left|-\right\rangle && (3.30)\\
&= \left(2\left|\varphi\right\rangle - \left|\varphi\right\rangle - 2\cdot\frac{2}{\sqrt{2^n}}\cdot\frac{1}{\sqrt{2^n}}\left|\varphi\right\rangle + \frac{2}{\sqrt{2^n}}\left|i_0\right\rangle\right)\left|-\right\rangle && (3.31)\\
&= \left(\frac{2^n-4}{2^n}\left|\varphi\right\rangle + \frac{2}{\sqrt{2^n}}\left|i_0\right\rangle\right)\left|-\right\rangle && (3.32)
\end{aligned}$$

em que $\left\langle\varphi\right|\varphi\left\rangle = 1\right.$ e $\left\langle\varphi\right|i_0\left\rangle = \frac{1}{\sqrt{2^n}}\left[\left\langle 0\right|i_0\left\rangle + \ldots + \left\langle i_0\right|i_0\left\rangle + \ldots + \left\langle 2^n-1\right|i_0\right\rangle\right] = \frac{1}{\sqrt{2^n}}\left[0 + \ldots + 1 + \ldots 0\right] = \frac{1}{\sqrt{2^n}}\right.$.

O estado $\left|\psi_3\right\rangle$ é o resultado da primeira iteração de Grover. Neste ponto, a probabilidade de medir o elemento procurado é dada por

$$p(i_0) = \left|\frac{2^n-4}{2^n}\cdot\frac{1}{\sqrt{2^n}} + \frac{2}{\sqrt{2^n}}\right|^2 \tag{3.33}$$

Quando $n > 1$, a probabilidade $p(i_0)$ é maior que $p(i \neq i_0)$, ou seja, é mais provável medir i_0 na saída do algoritmo do que medir os demais elementos, os quais não são soluções para o problema. Para que esta probabilidade torne-se próxima de 100%, para que se obtenha com alta probabilidade o valor i_0 na saída do algoritmo, sucessivas aplicações das iterações de Grover se fazem necessárias.

O operador G, que sintetiza as iterações de Grover, é dado por:

$$G = \left[(2\left|\varphi\right\rangle\left\langle\varphi\right| - \mathbb{I}) \otimes \mathbb{I}\right]U_f \tag{3.34}$$

e o número k das aplicações sucessivas da iteração de Grover é discutido a seguir.

Número Requerido de Iterações

Por meio de repetições da aplicação do operador G, o algoritmo de quântico de busca aproxima-se da solução desejada com certa probabilidade. Porém, para fins práticos, é necessário determinar um número ótimo de iterações que maximize a probabilidade de obtenção do resultado desejado.

O melhor número de iterações de Grover é definido por meio da análise dos operadores como rotações e, como resultado, o valor k do número de iterações é dado por:

$$k = \left\lfloor \frac{\pi}{4}\sqrt{2^n} \right\rceil \tag{3.35}$$

em que n é a quantidade de qubits no primeiro registrador e $\lfloor \cdot \rceil$ denota a função inteiro mais próximo. A obtenção deste valor k pode ser vista detalhadamente na obra de Nielsen e Chuang [2005].

3.1.3 Interpretação Geométrica

Além das expressões algébricas para explicar o algoritmo de busca quântica, existe também uma interpretação geométrica. Por ser uma representação visual, esta interpretação geométrica permite uma melhor compreensão de como os passos do algoritmo conduzem à solução correta. É também considerada um ferramental teórico que auxilia na descrição do efeito dos operadores do algoritmo de busca quântica nos estados de entrada.

Esta seção apresenta uma visão resumida da interpretação geométrica para o algoritmo de busca quântica, que pode ser vista com mais detalhes na obra de Portugal e colaboradores [2004]. Para prosseguir com tal representação, sejam os subespaços ortogonais gerados por $|i_0\rangle$ e $|u\rangle$, em que:

$$|u\rangle = \frac{1}{\sqrt{2^n - 1}} \sum_{i=0, i\neq i_0}^{2^n-1} |i\rangle \tag{3.36}$$

Como visto, após a aplicação das portas de Hadamard o estado do sistema é denotado por $|\psi_1\rangle$. Este estado possui componentes nos subespaços $|u\rangle$ e $|i_0\rangle$. O resultado da aplicação de U_f a este estado, permite uma interpretação geométrica da atuação deste operador no primeiro registrador: representa uma reflexão do vetor $|\psi_1\rangle$ em relação ao subespaço ortogonal

a $|i_0\rangle$, gerado por todos os outros elementos da base computacional. O resultado da aplicação de U_f é mostrado na Figura 3.3.

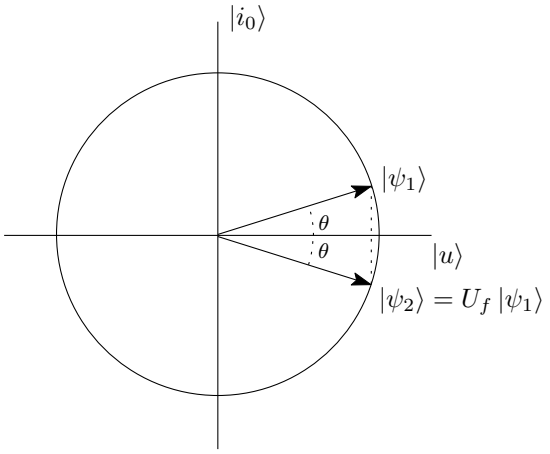

Figura 3.3: Representação geométrica da operação $|\psi_2\rangle = U_f\,|\psi_1\rangle$.

Seguindo esta representação, é necessário considerar o operador $2\,|\varphi\rangle\,\langle\varphi| - \mathbb{I}$, responsável pela amplificação de amplitude. Uma interpretação geométrica da atuação deste operador é de uma projeção de $|\psi_2\rangle$ sobre $|\psi_1\rangle$, resultando no aumento da amplitude de $|i_0\rangle$ no estado resultante ($|\psi_3\rangle$) quando comparado à sua amplitude no estado anterior ($|\psi_2\rangle$). A Figura 3.4 ilustra este processo.

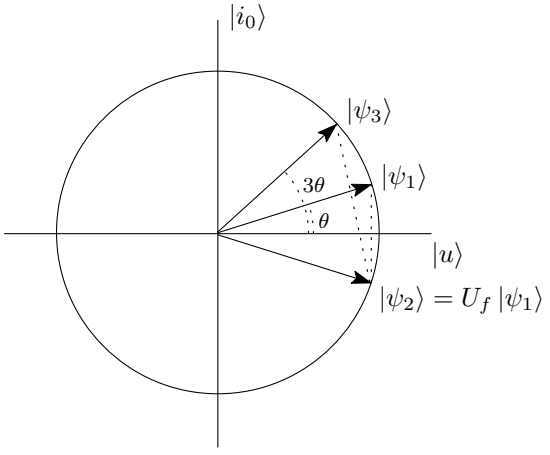

Figura 3.4: Representação geométrica da projeção de $|\psi_2\rangle$ em relação a $|\psi_1\rangle$.

Considerando que duas iterações de Grover são adequadas para o caso em questão, a Figura 3.5 exemplifica o passo-a-passo dos estados que o sistema assume durante estas iterações, exemplificando a interpretação geométrica para este algoritmo. O estado resultante,

denotado por $|\psi_5\rangle$, possui amplitude bastante próxima do elemento procurado, ou seja, o resultado da busca indica o elemento correto com, aproximadamente, 100% de certeza.

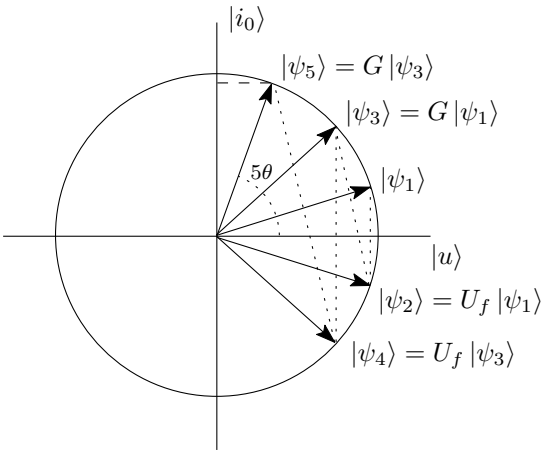

Figura 3.5: Visualização geométrica do operador de amplificação da amplitude.

Como mostrado na Seção A.2.1, há uma representação geométrica para qubits. No escopo do algoritmo de Grover, a entrada pode ser representada geometricamente da seguinte forma:

$$|\psi\rangle = \sin(\theta)\,|i_0\rangle + \cos(\theta)\,|u\rangle \tag{3.37}$$

em que $\sin^2(\theta) = \frac{1}{2^n}$ e $\cos^2(\theta) = \frac{2^n-1}{2^n}$.

Considerando esta representação da entrada e a interpretação geométrica do algoritmo de Grover, o operador deste algoritmo também pode ser denotado da seguinte forma, em que k representa o número de iterações necessárias:

$$G^k\,|\psi\rangle = \sin\left[(2k+1)\theta\right]|i_0\rangle + \cos\left[(2k+1)\theta\right]|u\rangle \tag{3.38}$$

Esta representação é bastante útil, pois além de prover outra interpretação, promove uma simplificação nos cálculos a serem realizados.

3.1.4 Múltiplas Soluções

Diz-se que uma busca possui múltiplas soluções se mais de um valor satisfaz à uma função binária f capaz de reconhecer diferentes soluções para este problema. Formalmente, a função f possui a seguinte definição:

$$f(x) = \begin{cases} 1, & \text{se } x \in \{i_0, i_1, \ldots, i_k\} \\ 0, & \text{caso contrário.} \end{cases} \tag{3.39}$$

em que $\{i_0, i_1, \ldots, i_k\} \subset \{0, 1, \ldots, 2^n - 1\}$.

Na proposição original do algoritmo de busca quântica, Grover considera apenas o caso de solução única [Grover 1997], embora seja possível adaptar este algoritmo para o caso de múltiplas soluções. Tal possibilidade de extensão revela que o algoritmo de Grover é o caso particular de um procedimento mais geral, denominado *amplificação de amplitude* [Chen et al. 2001; Kaye et al. 2007].

Para o caso de múltiplas soluções, considera-se que após a aplicação da porta de Hadamard, a entrada é particionada em dois subespaços: um subespaço de soluções ("bom") e um subespaço de não soluções ("ruim"), da seguinte forma, considerando uma interpretação geométrica da entrada:

$$|\psi\rangle = \sin(\theta)\,|\psi_{bom}\rangle + \cos(\theta)\,|\psi_{ruim}\rangle \tag{3.40}$$

Aplicações do operador de Grover nesta esta entrada são descritas por:

$$G^k\,|\psi\rangle = \sin[(2k+1)\theta]\,|\psi_{bom}\rangle + \cos[(2k+1)\theta]\,|\psi_{ruim}\rangle \tag{3.41}$$

em que k é o número de iterações adequadas, dado pela seguinte fórmula:

$$k = \left\lfloor \frac{\pi}{4}\sqrt{\frac{2^n}{M}} \right\rceil \tag{3.42}$$

em que n é a quantidade de qubits no primeiro registrador, M é a quantidade de soluções existentes e $\lfloor \cdot \rceil$ denota a função inteiro mais próximo.

3.1.5 Considerações sobre o Algoritmo Quântico de Busca

Para concluir a apresentação do algoritmo quântico de busca, algumas considerações sobre o mesmo são apresentadas.

A primeira destas considerações refere-se ao fato do algoritmo de busca quântica ser *ótimo*. Isto significa que qualquer algoritmo que acesse uma base de dados desordenada utilizando o operador U_f deve aplicá-lo, no mínimo, tantas vezes quanto o algoritmo de

busca quântica proposto por Grover o faz [Bennett et al. 1997]. O número de iterações de Grover a serem efetuadas também é provadamente ótimo [Boyer et al. 1998]. Estes dois resultados evidenciam que o melhor ganho para este tipo de problema é dado pelo algoritmo quântico de busca e que melhorias não são possíveis, considerando o paradigma em questão [Nielsen and Chuang 2005; Zalka 1998; 1999].

A segunda consideração diz respeito a aceleração de problemas $\mathcal{NP}$-completos. A partir de cada problema desta complexidade é possível definir um método de busca, que pode ser implementado com o algoritmo quântico de busca, capaz de encontrar uma solução para o problema em questão. Nestas condições, o ganho alcançado com o uso do algoritmo quântico de busca é quadrático [Nielsen and Chuang 2005; Hirvensalo 2001]. Este resultado permite a utilização do algoritmo quântico de busca em problemas de difícil solução, provendo ganhos em relação aos algoritmos clássicos equivalentes.

3.2 Algoritmo Quântico para o Logaritmo Discreto

A hipótese de intratabilidade computacional do problema do Logaritmo Discreto é base para a segurança de diversos sistemas de criptografia. Em virtude disto, diferentes estratégias para resolver este problema de forma eficiente têm sido objeto de estudo, pois podem implicar na quebra da segurança de diversos sistemas utilizados atualmente [Gregg 2003].

O problema do logaritmo discreto é fundamentado na Teoria dos Números e consiste em, dados g e $x = g^y$, obter o valor y, ou seja, efetuar o logaritmo discreto de g^y na base g. Em uma definição mais formal, o problema recebe como entrada um primo p, um gerador g de $\mathbb{Z}_p^*$, e $x \in \mathbb{Z}_p^*$, tal que $x = g^y$, e deve fornecer como saída y, o logaritmo discreto de x na base g módulo p.

Dentre os algoritmos clássicos mais utilizados para resolver o problema do logaritmo discreto, destacam-se os métodos *baby-step giant step*, *index calculus* e ρ-Pollard. Porém, todos eles são de ordem exponencial, ou seja, não resolvem o problema de modo eficiente [Gregg 2003]. Isto significa que há uma hipótese de intratabilidade do problema do logaritmo discreto perante computadores clássicos.

Diferentemente da Computação Clássica, o problema do logaritmo discreto é resolvido de modo eficiente por computadores quânticos, graças a um algoritmo proposto por Shor

[1997]. Este algoritmo é objeto de estudo das próximas seções. Porém, para que haja uma maior clareza antes desta apresentação, alguns conceitos matemáticos essenciais serão recordados.

Desta forma, a apresentação deste algoritmo é organizada como segue: inicialmente será apresentada a Transformada Quântica de Fourier, pois caracteriza um dos elementos essenciais do algoritmo quântico para o problema do logaritmo discreto. Em seguida, é apresentado o oráculo necessário para o algoritmo do logaritmo discreto. Os passos que compõem o algoritmo em questão são apresentados, em termos do circuito quântico correspondente. Por fim, é discutido o número de repetições requeridas para encontrar a solução correta com alta probabilidade.

3.2.1 Transformada Quântica de Fourier

A transformação de dados consiste de uma das práticas mais adotadas pela Ciência da Computação e pela Matemática para resolução de problemas. Uma das transformações mais conhecidas denomina-se *Transformada de Fourier*, com diversas aplicações, especialmente na área de Processamento Digital de Sinais.

A Transformada de Fourier clássica toma como entrada um vetor de número complexos $\vec{x} = (x_0, x_1, \ldots, x_{N-1})$, em que N é um parâmetro pré-fixado, e produz um vetor $\vec{y} = (y_0, y_1, \ldots, y_{N-1})$, em que cada y_k, $k = 0, 1, \ldots, N-1$, tem a seguinte forma:

$$y_k \equiv \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \cdot e^{\frac{2 \cdot \pi \cdot i \cdot j \cdot k}{N}} \tag{3.43}$$

A *Transformada Quântica de Fourier* (QFT – *Quantum Fourier Transform*) é o análogo quântico à Transformada de Fourier clássica. É implementada por um operador unitário que toma os elementos de uma base ortogonal $|0\rangle, |1\rangle, \ldots, |N-1\rangle$ e produz a seguinte transformação:

$$|j\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{\frac{2 \cdot \pi \cdot i \cdot j \cdot k}{N}} |k\rangle \tag{3.44}$$

Considerando $N = 2^n$, em que n é um inteiro, e a base $|0\rangle, |1\rangle, \ldots, |2^n - 1\rangle$ como a base computacional, é possível denotar a QFT de uma outra forma, comumente utilizada em diversos algoritmos quânticos. Denotando o estado $|j\rangle$ utilizando a notação binária, tem-se

$j = j_1 j_2 \dots j_n$. Mais formalmente, $j = j_1 \cdot 2^{n-1} + j_2 \cdot 2^{n-2} + \dots + j_n \cdot 2_0$. A seguinte notação pode ser utilizada para representar uma fração binária $j = 0.j_l j_{l+1} \dots j_m$, uma vez que $j = \frac{j_l}{2} + \frac{j_{l+1}}{4} + \dots + \frac{j_m}{2^{m-l+1}}$.

Deste modo, a QFT pode ser denotada utilizando a representação de produto, como segue:

$$|j_1, j_2, \dots j_n\rangle \rightarrow \frac{(|0\rangle + e^{2\cdot\pi\cdot i\cdot 0.j_n} |1\rangle)(|0\rangle + e^{2\cdot\pi\cdot i\cdot 0.j_{n-1}j_n} |1\rangle) \dots (|0\rangle + e^{2\cdot\pi\cdot i\cdot 0.j_1 j_2 \dots j_n} |1\rangle)}{2^{\frac{n}{2}}} \tag{3.45}$$

Além da QFT, define-se a Transformada Quântica de Fourier Inversa, denotada por QFT^{-1}, que possui a seguinte atuação:

$$\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{\frac{2\cdot\pi\cdot i\cdot k\cdot j}{N}} |k\rangle \rightarrow \left|\frac{\tilde{j}}{N}\right\rangle \tag{3.46}$$

A inversa da QFT retorna apenas uma estimativa do valor de j, pois leva em consideração a melhor fração binária que aproxima este valor.

A QFT é um dos componentes do algoritmo quântico que resolve o problema do logaritmo discreto, a ser apresentado na próxima seção. Mais detalhes sobre aplicações, provas de corretude, complexidade e circuito quântico que implementa esta transformada podem ser encontrados nas obras de Nielsen e Chuang [2005] e de Kaye et al. [2007].

3.2.2 Oráculo para o Algoritmo do Logaritmo Discreto

O algoritmo quântico para o problema do logaritmo discreto faz uso de um oráculo que implementa uma função $f : \mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1} \rightarrow \mathbb{Z}_p^*$ com a seguinte definição:

$$f(a, b) = g^a \cdot x^{-b} \bmod p \tag{3.47}$$

Note que a função f tem por parâmetros os valores g e x, que são a base do logaritmo e o número do qual deseja-se saber o logaritmo discreto, respectivamente. Observa-se que a função f é computável em tempo polinomial e, uma vez que $y \equiv \log_g x$, esta função pode também ser denotada da seguinte forma:

$$f(a,b) = g^a \cdot (g^y)^{-b} \bmod p \tag{3.48}$$

$$= g^a \cdot g^{-y \cdot b} \bmod p \tag{3.49}$$

$$= g^{a-y \cdot b} \bmod p \tag{3.50}$$

É importante ressaltar que $f(a_1, b_1) = f(a_2, b_2)$ se, e somente se, $(a_2, b_2) = (a_1, b_1) + \lambda(y, 1)$, para qualquer $\lambda \in \mathbb{Z}_{p-1}$. Diz-se que o par ordenado $(y, 1)$ é o *período* da função f.

3.2.3 Apresentação do Algoritmo Quântico para o Logaritmo Discreto

A caracterização do algoritmo quântico para o logaritmo discreto apresentado nesta seção segue os passos descritos no artigo de Jozsa [2001] e na obra de Kaye et al. [2007]. Sugere-se que o leitor consulte as obras mencionadas caso queira expandir os seus conhecimentos sobre o referido algoritmo.

O algoritmo quântico que resolve o problema do logaritmo discreto faz uso de três registradores. Há também três tipos de portas utilizadas neste algoritmo: a porta de Hadamard, denotada por H; o oráculo U_f, cuja função que implementa é apresentada na Eq. (3.47); e a porta QFT, que implementa a Transformada Quântica de Fourier. Neste algoritmo, as medições acontecem em momentos diferentes, inicialmente no terceiro registrador e posteriormente nos dois primeiros. O circuito ilustrado na Figura 3.6 contempla os elementos apresentados para a resolução do problema do logaritmo discreto.

Para ilustrar como este algoritmo resolve o problema do logaritmo discreto, é necessário seguir os passos demandados pelo mesmo. Inicialmente, todos os registradores são inicializados com $|0\rangle$, como mostrado no estado inicial $|\psi_0\rangle$ a seguir:

$$|\psi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n} |0\rangle^{\otimes r} \tag{3.51}$$

O primeiro passo do algoritmo consiste na aplicação da porta de Hadamard aos dois primeiros registradores, colocando-os em uma superposição igualmente distribuída. O estado após este passo, $|\psi_1\rangle$, é mostrado a seguir:

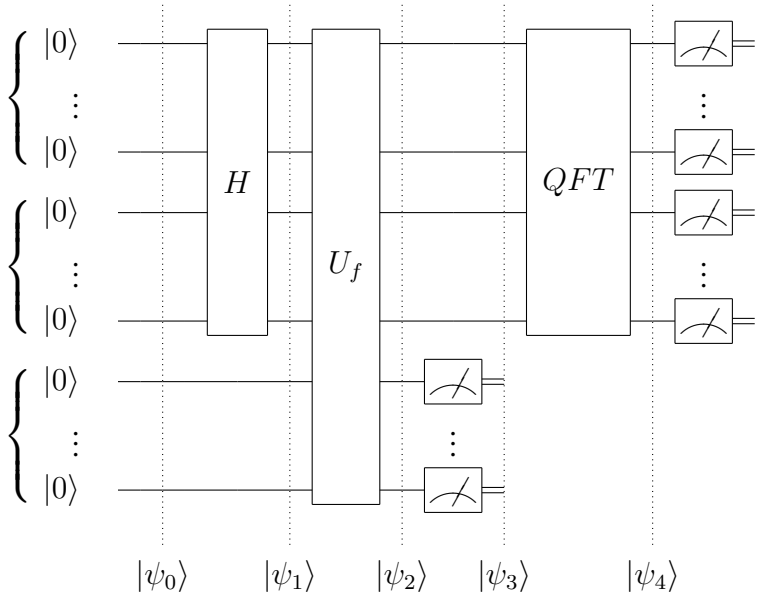

Figura 3.6: Circuito quântico que resolve o problema do logaritmo discreto.

$$|\psi_1\rangle = H^{\otimes 2}|\psi_0\rangle \tag{3.52}$$

$$= \frac{1}{\sqrt{2^n}} \cdot \frac{1}{\sqrt{2^n}} \sum_{a=0}^{2^n-1} |a\rangle \sum_{b=0}^{2^n-1} |b\rangle |0\rangle \tag{3.53}$$

$$= \frac{1}{2^n} \sum_{a=0}^{2^n-1} \sum_{b=0}^{2^n-1} |a\rangle |b\rangle |0\rangle \tag{3.54}$$

No escopo do logaritmo discreto, são interessantes apenas os valores de a e b que pertencem a $\mathbb{Z}_{p-1}$, pois este é o domínio da função f implementada pelo oráculo. Para os valores que não são de interesse, uma comparação com p ao final do processo pode ser realizada, implicando em uma nova execução do algoritmo caso o valor lido seja maior que p [Shor 1997]. Considerando apenas os valores de interesse, $|\psi_1\rangle$ pode ser denotado da seguinte forma:

$$|\psi_1\rangle = \frac{1}{p-1} \sum_{a=0}^{p-2} \sum_{b=0}^{p-2} |a\rangle |b\rangle |0\rangle \tag{3.55}$$

Para dar prosseguimento ao algoritmo quântico, é necessário aplicar a porta U_f, resultando em $|\psi_2\rangle$:

$$|\psi_2\rangle = U_f |\psi_1\rangle \tag{3.56}$$

$$= U_f \left(\frac{1}{p-1} \sum_{a=0}^{p-2} \sum_{b=0}^{p-2} |a\rangle |b\rangle |0\rangle \right) \tag{3.57}$$

$$= \frac{1}{p-1} \sum_{a=0}^{p-2} \sum_{b=0}^{p-2} |a\rangle |b\rangle |f(a,b)\rangle \tag{3.58}$$

Como mostra o circuito, uma medição deve ser realizada no terceiro registrador. Esta medição implica que a superposição existente irá colapsar e o resultado da medição será um dado $k_0 = f(a_0, b_0)$. Fazendo uso dos resultados da função f mostrados na Seção 3.2.2, o sistema entra no estado periódico $|\psi_3\rangle$:

$$|\psi_3\rangle = \frac{1}{\sqrt{p-1}} \sum_{k=0}^{p-2} |a_0\rangle |b_0\rangle + k \cdot |y\rangle |1\rangle \tag{3.59}$$

$$= \frac{1}{\sqrt{p-1}} \sum_{k=0}^{p-2} |a_0\rangle |b_0\rangle + |k \cdot y\rangle |k\rangle \tag{3.60}$$

$$= \frac{1}{\sqrt{p-1}} \sum_{k=0}^{p-2} |a_0 + k \cdot y\rangle |b_0 + k\rangle \tag{3.61}$$

Para eliminar de $|\psi_3\rangle$ a dependência dos valores (a_0, b_0) escolhidos aleatoriamente após a medição realizada no terceiro registrador, é necessário aplicar a Transformada de Fourier Quântica módulo $p-1$ nos registradores remanescentes. Porém, antes desta aplicação, algumas considerações são necessárias.

Seja uma função χ_{l_1,l_2} com a seguinte definição:

$$\chi_{l_1,l_2}(a,b) = e^{2 \cdot i \cdot \pi \frac{(a \cdot l_1 + b \cdot l_2)}{p-1}} \tag{3.62}$$

Após a transformada de Fourier, o sistema entrará em uma superposição de valores (l_1, l_2) tal que:

$$\chi_{l_1,l_2}(y,1) = e^{\frac{2 \cdot i \cdot \pi (y \cdot l_1 + 1 \cdot l_2)}{p-1}} = 1 \tag{3.63}$$

isto é, $y \cdot l_1 + l_2 \equiv 0 \bmod p-1$. Deriva-se também que $l_2 = -y \cdot l_1$ para $l_1 = 0, 1, \cdot p-2$.

Dando prosseguimento ao algoritmo e fazendo uso das considerações apresentadas, o estado $|\psi_4\rangle$ pode ser denotado por:

$$|\psi_4\rangle = QFT^{\otimes 2} |\psi_3\rangle \tag{3.64}$$

$$= QFT^{\otimes 2} \left(\frac{1}{\sqrt{p-1}} \sum_{k=0}^{p-2} |a_0 + k \cdot y\rangle \, |b_0 + k\rangle \right) \tag{3.65}$$

$$= \frac{1}{\sqrt{p-1}} \sum_{l_1=0}^{p-2} e^{\frac{2 \cdot i \cdot \pi}{p-1} \cdot (a_0 \cdot l_1 - b_0 \cdot y \cdot l_1)} \, |l_1\rangle \, |-y \cdot l_1\rangle \tag{3.66}$$

Para concluir o último passo do algoritmo, medições devem ser realizadas nos dois primeiros registradores e duas situações devem ser consideradas:

1. Quando l_1 é co-primo de p: Neste caso, encontrar l_1^{-1}, o inverso multiplicativo de l_1 módulo $p-1$. Por fim, multiplicar l_1^{-1} pelo resultado da medição no segundo registrador, obtendo y;

2. Quando l_1 não é co-primo de p: Esta situação indica uma falha no resultado obtido. O algoritmo deve ser executado novamente.

Diante da possibilidade de ocorrência de falhas, é necessário executar o algoritmo algumas vezes para obter confiança nos resultados. O número de repetições necessárias é discutido na seção a seguir.

Número de Repetições Necessárias

O número de repetições necessárias do algoritmo quântico do logaritmo discreto está ligada à probabilidade do valor l_1 lido ao final do processo ser ou não co-primo de p.

De acordo com a Teoria dos Números, a quantidade de co-primos menores que um dado valor p é dada por $e^{-\gamma} p / \log\log p$, dado p grande e γ a constante de Euler. Assim, a probabilidade de um valor λ qualquer ser co-primo de p é dada por $O(1/\log\log p)$. Isto implica que um número de repetições da ordem de $O(\log\log p)$ faz com que a probabilidade do valor lido ser co-primo de p seja próxima de 1. Portanto, $O(\log\log p)$ é o número de repetições necessárias [Jozsa 2001].

É interesse salientar que o número de repetições necessárias é de ordem polinomial e, juntamente com os outros componentes, a complexidade resultante do algoritmo quântico em questão é da ordem de $O(n^3)$, em que n é número de bits para representar p. Em suma,

trata-se de um algoritmo eficiente para o problema do logaritmo discreto. Em relação aos algoritmos clássicos equivalentes, o algoritmo quântico proposto por Shor apresenta um ganho superpolinomial [Kaye et al. 2007].

Notas do Capítulo

Neste capítulo foram apresentados dois algoritmos quânticos: o algoritmo quântico para busca em base de dados desordenada e o algoritmo quântico para o problema do logaritmo discreto.

Em relação ao algoritmo quântico de busca, foi visto que este algoritmo opera inicialmente colocando os estados de entrada em superposição e, posteriormente, invertendo e amplificando a fase do elemento procurado em detrimento das demais amplitudes. Após este processo, uma medição retorna, com grande probabilidade, o elemento procurado, ou seja, encontra uma solução para o problema de busca. Foi visto que este algoritmo resolve o problema de busca com ganho quadrático em relação a sua contrapartida clássica.

O algoritmo quântico para o logaritmo discreto, por sua vez, faz uso da transformada de Fourier, de um oráculo e da porta de Hadamard para resolver o problema do logaritmo discreto com alta probabilidade. Neste algoritmo, são necessárias medições em momentos diferentes, para que a superposição existente seja colapsada para um conjunto específico de valores antes do final da computação, conforme apresentado. Para garantir confiança no resultado do algoritmo, é necessário repeti-lo um número polinomial de vezes. Este algoritmo quântico é eficiente, da ordem de $O(n^3)$, e representa um ganho superpolinomial em relação ao melhor algoritmo clássico conhecido para este problema.

Ambos os algoritmos ilustram a utilização da Computação Quântica na resolução de problemas, possuem importância do ponto de vista prático e consolidam ganhos em relação ao paradigma da Computação Clássica.

Capítulo 4

Ataque Quântico ao Gerador de Blum-Micali

Com o intuito de responder às questões de pesquisa motivadoras deste trabalho, este capítulo apresenta a proposição de um ataque quântico ao gerador de Blum-Micali, amplamente utilizado em aplicações criptográficas. Neste ataque, um adversário, em posse de um computador quântico e de bits interceptados do gerador, torna-se capaz de reproduzir saídas deste CSPRNG, comprometendo completamente a sua imprevisibilidade.

Para apresentar o ataque proposto, este capítulo está organizado como segue: na Seção 4.1 é caracterizado o gerador Blum-Micali, sua definição e uma representação por meio de grafos funcionais. Após esta definição, um ataque clássico para este gerador é apresentado na Seção 4.2, juntamente com a classificação do mesmo de acordo com a taxonomia de Kelsey, análise da quantidade de bits requeridos e determinação da complexidade computacional, detalhados na Seção 4.3. Um algoritmo quântico com melhor eficiência que o algoritmo clássico apresentado na Seção 4.2, é proposto na Seção 4.4. A análise deste ataque é delineada na Seção 4.5.

Este capítulo apresenta as contribuições deste trabalho no que diz respeito a possibilidade e viabilidade de utilizar o paradigma da Computação Quântica na formulação de ataques a geradores, com ganhos em relação aos algoritmos clássicos equivalentes.

4.1 Gerador de Blum-Micali

O gerador pseudo-aleatório de Blum-Micali (BM) foi o primeiro gerador criptograficamente seguro a ser proposto na literatura [Blum and Micali 1984]. Este gerador é composto de dois elementos: (i) uma permutação *one-way* que implementa a função recursiva do gerador; e (ii) um predicado difícil para a permutação *one-way*, o qual produz os bits que serão apresentados na saída do gerador. A segurança do gerador de Blum-Micali é baseada na hipótese de intratabilidade da sua permutação *one-way*, a qual se baseia no problema do logaritmo discreto [Sidorenko 2008; Sidorenko and Schoenmakers 2005]

Alguns conceitos matemáticos são necessários para que se entenda o funcionamento deste gerador. Seja p um primo grande e $n = \lceil \log p \rceil$ o comprimento de p em bits. O conjunto $\mathbb{Z}_p^* = \{1, 2, \ldots, p-1\}$ denota o grupo cíclico sob a multiplicação $\bmod\ p$. Seja g um gerador de $\mathbb{Z}_p^*$ e $x_0 \in_R \mathbb{Z}_p^*$, em que $a \in_R A$ denota a escolha aleatória de um elemento a em um conjunto A.

O gerador de Blum-Micali é preparado com os parâmetros (p, g, x_0), como previamente descritos, em que x_0 é a semente deste gerador. O gerador de Blum-Micali produz bits pseudo-aleatórios utilizando a permutação *one-way* do logaritmo discreto sobre o domínio $\mathbb{Z}_p^*$ e um predicado difícil δ, como mostrados a seguir:

$$x_i = g^{x_{i-1}} \bmod p \tag{4.1}$$

$$b_i = \delta(x_i) \tag{4.2}$$

A Eq. (4.1) é denominada *mapa exponencial* e a função δ possui a seguinte definição:

$$\delta(x) = \begin{cases} 1, & \text{se } x > \frac{p-1}{2} \\ 0, & \text{caso contrário} \end{cases} \tag{4.3}$$

No gerador de Blum-Micali, os parâmetros p e g estão disponíveis publicamente. Por definição, o estado interno do gerador Blum-Micali, bem como a semente, devem ser mantidos em segredo. Somente os bits produzidos pela função δ podem ser vistos diretamente.

A segurança deste gerador baseia-se na dificuldade de inverter a função $x_{i+1} = g^{x_i} \bmod p$ em $x_i = \log_g(x_{i+1}) \bmod p$. Esta inversão é equivalente a resolver

instâncias do problema do logaritmo discreto, mas é importante ressaltar que não se conhece nenhum algoritmo clássico eficiente capaz de realizar esta operação.

Para ilustrar a utilização do gerador Blum-Micali, suponha que este foi configurado com os parâmetros $(7, 3, 1)$, denotando p, g e x_0, respectivamente. A Figura 4.1 mostra um diagrama que descreve a evolução do estado interno deste gerador e os bits produzidos ao longo deste processo.

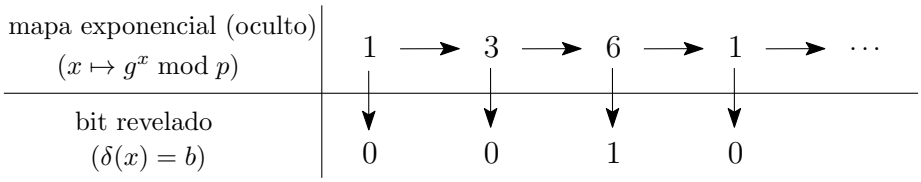

Figura 4.1: Diagrama do funcionamento de um gerador Blum-Micali inicializado com parâmetros $(p = 7, g = 3, x_0 = 1)$.

4.1.1 Grafo Funcional do Gerador de Blum-Micali

Qualquer gerador de Blum-Micali também pode ser denotado em termos de seus *grafos funcionais*. Um grafo funcional é um dígrafo que respeita a restrição de haver apenas uma única aresta saindo de cada vértice [Cloutier and Holden 2010]. Tal representação gráfica auxilia na compreensão do funcionamento e dos procedimentos de inicialização deste gerador.

O grafo funcional de um gerador Blum-Micali, denotado por $G = \langle V, E \rangle$, é um grafo cujo conjunto de vértices é dado por $V = \mathbb{Z}_p^*$. Há uma aresta $(x, y) \in E$, dirigida do vértice x para o vértice y, sempre que há um mapeamento exponencial $x \mapsto y$ (ver Eq. (4.1)). Tal aresta é rotulada com o bit produzido pelo vértice de origem, isto é, com $\delta(x)$.

Para ilustrar tal tipo de grafo, a Figura 4.2 mostra o grafo do gerador de Blum-Micali inicializado com parâmetros $(p = 7, g = 3, x_0 = 1)$. O vértice 1, relativo a semente, encontra-se enfatizado.

De acordo com esta figura, é possível identificar a existência de três auto-laços, isto é, três vértices cujas arestas são dirigidas para o próprio vértice, e também a existência de um ciclo de comprimento 3, contendo os vértices 1, 3 e 6. Os vértices com auto-laço são denominados *pontos fixos* no mapa exponencial do gerador de Blum-Micali. Estes pontos fixos devem ser evitados como semente, pois produzem um único tipo de bit na saída do gerador, tornando-o

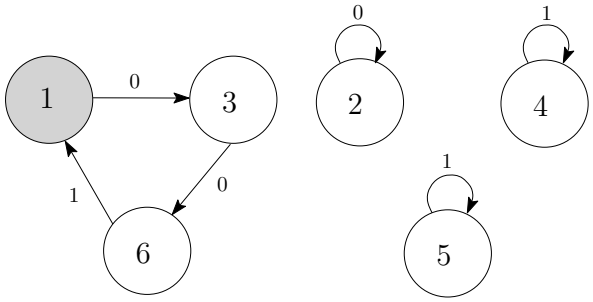

Figura 4.2: Grafo funcional do gerador de Blum-Micali com parâmetros $(p = 7, g = 3, x_0 = 1)$.

previsível. Mais além, a escolha da semente do gerador deve levar em conta os ciclos mais longos, com o intuito de maximizar o período do gerador.

Devido a adoção deste gerador em sistemas criptográficos, é de crucial importância analisar a vulnerabilidade do mesmo contra ataques. A próxima seção apresenta um ataque clássico ao gerador de Blum-Micali, que faz uso de bits interceptados da saída do gerador. As provas de corretude e uma análise de complexidade do ataque também são apresentadas.

4.2 Ataque Clássico ao Gerador de Blum-Micali

Nesta seção será descrito um ataque clássico ao gerador de Blum-Micali. Para apresentar este ataque, suponha a existência de um adversário do gerador. Tal adversário conhece os parâmetros públicos p e g, e também interceptou uma seqüência finita de j bits da saída do gerador, denotada por $\mathbf{b}(j) = \{b_i\}$, $i = 1, 2, \ldots, j$. O adversário tem interesse de recuperar o estado interno do gerador, i.e., o conjunto $X(j)$.

Para obter o estado interno do gerador, o adversário deve utilizar os bits em $\mathbf{b}(j)$ e a informação de que $x_0 \in \mathbb{Z}_p^*$. Com os parâmetros públicos, é possível remontar as regras (4.1) e (4.2). Assim, o adversário precisa fazer suposições sobre os elementos em $X(j)$ e utilizar os bits em $\mathbf{b}(j)$ para confirmá-las ou refutá-las.

O conjunto $\hat{X}_i$ será referido como o *conjunto de estimadores relativos a* x_i, ou simplesmente, *conjunto de estimadores*. Para realizar o ataque ao gerador de Blum-Micali, o adversário iniciará com $\hat{X}_0 = \mathbb{Z}_p^*$, uma vez que $x_0 \in \mathbb{Z}_p^*$.

Dados um conjunto de estimadores $\hat{X}_{i-1}$ e o bit $b_i \in \mathbf{b}(j)$, o adversário irá proceder da

seguinte forma para obter $\hat{X}_i$. Inicialmente irá computar $A_i = g^{\hat{X}_{i-1}} \bmod p$, ou seja, irá obter o conjunto A_i a partir de todos os elementos de $\hat{X}_{i-1}$ sob a ação do mapa exponencial. Seja $A_i = A_i^{(0)} \cup A_i^{(1)}$ uma partição do conjunto A_i devido a regra δ (ver Eq. (4.2)), i.e., $a \in A_i^{(0)}$ se, e somente se, $\delta(a) = 0$, e de modo análogo para $A_i^{(1)}$. Deste modo, o conjunto de estimadores $\hat{X}_i$ será dado por:

$$\hat{X}_i = A_i^{(b_i)} \tag{4.4}$$

É importante enfatizar que $\hat{X}_i$, $i = 1, 2, \ldots, j$, somente contém elementos da classe definida por $A_i^{(b_i)}$. Isto reduz a incerteza que o adversário tem sobre os elementos em $X(j)$, uma vez que a cardinalidade do conjunto de estimadores tende a diminuir a cada passo. Este procedimento resulta na obtenção de $x_j \in X(j)$. Em posse do representante x_j, o adversário pode obter os demais elementos de $X(j)$ por meio de aplicações de um algoritmo para o logaritmo discreto. A síntese do procedimento de recuperação do representante é descrita no Algoritmo 4.1.

Algoritmo 4.1 Algoritmo clássico de ataque ao gerador Blum-Micali – Síntese dos passos até a obtenção do representante. Considera-se que a todo conjunto quando instanciado encontra-se vazio

$i \leftarrow 1$
$\hat{X}_0 \leftarrow \{1, 2, \ldots, p-1\}$
while $(i \leq j)$ **do**
  for all $x \in \hat{X}_{i-1}$ **do**
    if $\delta(g^x \bmod p) = b_i$ **then**
      $\hat{X}_i \leftarrow \hat{X}_i \cup \{g^x \bmod p\}$
    end if
  end for
end while
print $\hat{X}_j$

Para verificar que o ataque descrito recupera o estado interno do gerador Blum-Micali é necessário provar que: (i) cada conjunto de estimadores $\hat{X}_i$ contém $x_i \in X(i)$; e que (ii) dados suficientes bits em $\mathbf{b}(j)$, a cardinalidade do conjunto $\hat{X}_i$ tende a 1 para i suficientemente grande ou então o próximo bit é completamente previsível. Para eliminar trivialidades, supõe-se que x_0 pertence a um ciclo longo no grafo funcional do gerador. Esta hipótese é levada em consideração, pois, do contrário, haveria grande previsibilidade da seqüência produzida pelo gerador. As provas são apresentadas a seguir.

Lema 4.2.1. *Cada conjunto de estimadores $\hat{X}_i$ contém $x_i \in X(i)$.*

Prova. *Indução em i. Claramente, a afirmação é válida para $i = 0$ dado que $x_0 \in \hat{X}_0 = \mathbb{Z}_p^*$. Assuma que $x_i \in \hat{X}_i$ (hipótese indutiva). Deve ser provado que $x_{i+1} \in \hat{X}_{i+1}$. De acordo com a Eq. (4.4), um certo $y \in \hat{X}_{i+1}$ se, e somente se: (1) $y = g^x \bmod p$ para algum $x \in \hat{X}_i$; e (2) $\delta(y) = b_{i+1}$. De acordo com a hipótese indutiva, $x_i \in \hat{X}_i$ e, conforme as produções do gerador, $\delta(x_{i+1} = g^{x_i} \bmod p) = b_{i+1}$. Assim, x_{i+1} atende a ambos os requisitos. Deste modo, a prova está concluída.* □

Fazendo uso do ataque clássico apresentado, o adversário é capaz de predizer o bit em duas situações: (i) $\left|\hat{X}_i\right| = 1$, e (ii) $A_{i+1}^{(0)} = \emptyset$ ou $A_{i+1}^{(1)} = \emptyset$. A primeira situação acontece quando o número de bits observado pelo adversário foi suficiente para identificar com precisão o representante do estado interno do gerador. Nesta situação, não há dúvidas de qual será o próximo bit produzido, pois basta o adversário realizar o mapa exponencial do representante do estado interno já identificado. A segunda situação permite que o adversário seja capaz de predizer o próximo bit corretamente, embora não conheça qualquer elemento do estado interno do gerador. Em outras palavras, na primeira situação não há incerteza sobre o estado interno do gerador nem sobre o próximo bit a ser produzido. Na segunda situação, não há incerteza sobre o próximo bit a ser produzido, embora haja incerteza sobre o estado interno do gerador.

Considerando i grande e que a segunda situação é menos provável de acontecer, dada a hipótese levantada previamente, o adversário irá olhar bits e reduzir o conjunto de estimadores a cada passo. A convergência para uma solução é sintetizada na prova a seguir.

Lema 4.2.2. *A cardinalidade dos conjuntos de estimadores vai diminuindo, isto é:*

$$|\hat{X}_{i+1}| \leq |\hat{X}_i|, \quad i = 1, 2, \ldots \tag{4.5}$$

Se há igualdade na Eq. (4.5) , então o bit b_{i+1} é completamente previsível.

Este lema esclarece a discussão a respeito da incerteza sobre os bits produzidos pelo gerador, que só é mantida caso os estimadores diminuam a cada passo.

Prova.

$$\left|\hat{X}_{i+1}\right| \overset{(a)}{=} \left|A_{i+1}^{(b_{i+1})}\right| \overset{(b)}{\leq} |A_{i+1}| \overset{(c)}{=} \left|g^{\hat{X}_i}\right| \pmod p \overset{(d)}{=} \left|\hat{X}_i\right|. \tag{4.6}$$

em que as igualdades (a) e (c) correspondem às definições de $\hat{X}_{i+1}$ e de A_{i+1}, respectivamente; a igualdade (b) refere-se ao fato de $A_{i+1}^{(b_{i+1})}$ ser um subconjunto de A_{i+1} e de ambos serem conjuntos finitos; e, por fim, a igualdade (d) deriva de g^x ser um mapa bijetivo. Isto conclui a primeira parte. Para mostrar que b_{i+1} é completamente conhecido quando há igualdade na Eq. (4.5), assume-se que $|\hat{X}_{i+1}| = |\hat{X}_i|$ é verdade na Eq. (4.6). Segue-se, portanto, que $|A_{i+1}^{(b_{i+1})}| = |A_{i+1}|$. Uma vez que $A_{i+1}^{(b_{i+1})} \subset A_{i+1}$, a igualdade só é possível somente se $A_{i+1}^{(b_{i+1})} = A_{i+1}$. Assim, o adversário é capaz de prever o próximo bit, uma vez que A_{i+1} só contém elementos da classe x tal que $\delta(x) = b_{i+1}$. □

Para exemplificar o ataque descrito, suponha que o adversário interceptou 2 bits seqüenciais, $\mathbf{b}(2) = \{10\}$, da saída de um gerador Blum-Micali com parâmetros públicos $p = 7$ e $g = 3$. O adversário inicia o ataque com um conjunto de estimadores para a semente $\hat{X}_0 = \{1, 2, \ldots, 6\}$. O adversário aplica o mapa exponencial e, em seguida, com a informação que o primeiro bit observado foi $b_1 = 1$, descarta os bits menores que $(7 - 1)/2 = 3$. Estes passos resultam em um conjunto de estimadores para x_1, $\hat{X}_1 = \{4, 5, 6\}$. Procedendo da mesma maneira, mas com o próximo bit observado $b_2 = 0$, o adversário obtém, a partir de $\hat{X}_1$, os estimadores para x_2, $\hat{X}_2 = \{1\}$. Neste caso, com apenas 2 bits interceptados, o adversário pôde recuperar, com 100% de certeza, o representante do estado interno do gerador. Para obter por completo o conjunto $X(2)$, o adversário deve realizar duas aplicações do logaritmo discreto ao valor obtido, resultado em $X(2) = \{x_2 = 1, x_1 = 6, x_0 = 3\}$. Uma abordagem gráfica para este exemplo é apresentada na Figura 4.3.

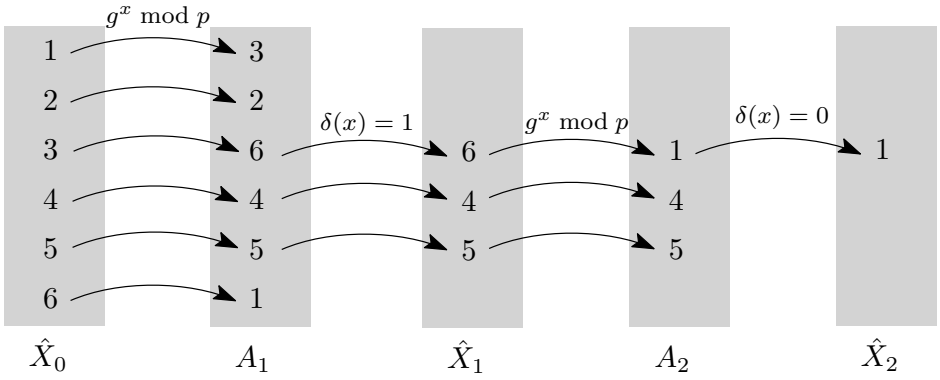

Figura 4.3: Representação gráfica de um ataque realizado a um gerador de Blum-Micali.

4.3 Análise do Ataque Clássico ao Gerador Blum-Micali

Nesta seção, o ataque clássico apresentado será analisado em detalhes. Esta análise compreende a classificação de acordo com a taxonomia de Kelsey, a determinação da quantidade de bits requeridos para a realização bem-sucedida de um ataque e, por fim, a determinação da complexidade do algoritmo. Tais elementos auxiliam na compreensão das características e limitações do algoritmo apresentado.

4.3.1 Classificação do Ataque de Acordo com a Taxonomia de Kelsey

O ataque realizado compromete, primeiramente, o representante do estado interno do gerador e, a partir deste, todo o estado interno do gerador. Com isto, é possível saber quais os bits já foram produzidos pelo gerador e também quais serão, a partir da aplicação do mapa exponencial. De acordo com a taxonomia de Kelsey, este ataque classifica-se como um *ataque de comprometimento permanente*, o qual recupera a semente e neutraliza a imprevisibilidade do gerador Blum-Micali.

4.3.2 Quantidade de Bits Requeridos

Levando em consideração que o período mais longo de um Gerador Blum-Micali possui $p-1$ elementos, um algoritmo ingênuo seria capaz de predizer a saída deste gerador após $p-1$ tentativas, ou analogamente, $p-1$ bits observados. A partir do Lema 4.2.2, observa-se que cada bit observado corresponde a um bit do representante do estado interno do gerador que passa a ser conhecido. Isto implica em um limitante de $\log p$ bits para que o adversário possa

determinar o representante do estado interno do gerador com alta probabilidade.

Para verificar este limitante, foi realizado um experimento no qual o adversário solicita tantos bits quantos forem necessários até que seja capaz de predizer a saída do gerador. Diversas repetições com diferentes configurações de geradores Blum-Micali foram utilizadas. As escolhas de (p, g, x_0) para estas configurações foram:

1. **Escolha do primo** p. Todos os números primos no intervalo $[257, 17863]$ foram considerados;

2. **Escolha do gerador** g. Quatro valores de g foram escolhidos: 3, 5, 17 e 19. A necessidade de utilizar diferentes valores para g advém da preocupação em eliminar algum viés inserido quando apenas um valor é fixado. Apenas as combinações em que g era um gerador de $\mathbb{Z}_p^*$ foram consideradas;

3. **Escolha da semente** x_0. Feita de forma aleatória com a utilização de um gerador de números pseudo-aleatórios com distribuição uniforme em $\mathbb{Z}_p^*$. Se x_0 caracterizava um ponto fixo no grafo funcional do gerador Blum-Micali, a escolha deste valor era refeita.

Após a conclusão de cada ataque, o número de bits demandado e o valor de p eram armazenados. Esta quantidade de bits requerida pelo adversário até predizer a saída do gerador com 100% de certeza foi a variável de resposta do experimento. A hipótese nula (H_0) do experimento era que o número de bits requeridos pelo adversário era limitado por $\log p$, e a hipótese alternativa (H_a) era de que este limitante não se mostrava válido. O nível de confiança escolhido foi de 95%. Para avaliar as hipóteses levantadas, um teste de média nula modificado foi realizado[1].

Os resultados observados foram sumarizados de acordo com o número de bits em p, definindo 6 grupos com valores de p de 8 a 14 bits. Estes resultados podem ser vistos na Tabela 4.1.

Para alcançar significância estatística em cada grupo, havia um número mínimo de amostras requeridas, descrito na Coluna *Amostras Requeridas*. Em todos os casos, este número

[1]Uma completa descrição sobre os testes de hipóteses e sobre os procedimentos de sumarização de dados utilizados neste experimento pode ser vista no livro de Jain [1991].

Tabela 4.1: Resultados experimentais agrupados de acordo com o número de bits em p.

Bits em p	Amostras Requeridas	Amostras Utilizadas	Média	σ	Mediana	Intervalo de Confiança
8	168	561	$8,044$	$1,333$	$8,000$	$(7,933, 8,155)$
9	351	453	$9,110$	$2,178$	$9,000$	$(8,909, 9,311)$
10	329	354	$10,107$	$2,341$	$10,000$	$(9,862, 10,3520)$
11	297	591	$11,090$	$2,441$	$11,000$	$(10,892, 11,286)$
12	241	1074	$12,011$	$2,379$	$12,000$	$(11,863, 12,158)$
13	196	1979	$13,111$	$2,533$	$13,000$	$(12,999, 13,222)$
14	183	1679	$14,008$	$2,420$	$14,000$	$(13,892, 14,124)$

foi respeitado, como reportado na Coluna *Amostras Utilizadas*. O número de bits utilizado pelo adversário foi sumarizado em termos de média, desvio padrão (denotado por σ) e mediana. Considerando que a média é similar ao número de bits em p e que o desvio padrão é pequeno, os dados observados em todos os ataques estão dentro de um escopo estreito. A mediana coincide com $\log p$ em todos os casos. Estes dados mostram uma pequena variação no número de bits requerido pelo adversário para realizar os ataques, considerando cada grupo.

A Coluna *Intervalo de Confiança* mostra os intervalos de confiança construídos a partir dos dados observados. É possível observar que o valor da quantidade de bits em p está inclusa no intervalo de cada grupo. Assim, de acordo com os testes realizados, é possível concluir, ao nível de significância de 95%, a ausência de evidências para rejeitar H_0 em todas as situações observadas. Mais ainda, uma vez que os intervalos são estreitos, não excedendo 1 bit, há a indicação de que a quantidade de bits demandada pelo adversário para atacar o gerador foi estimada com alto grau de precisão.

Embora os resultados do experimento tenham sido positivos, não é possível generalizá-los. Mais experimentos devem ser realizados, pois há infinitas maneiras para inicializar um gerador de Blum-Micali. Os resultados obtidos apenas reforçam a conformidade dos resultados teóricos apresentados.

4.3.3 Análise de Complexidade

Para analisar a complexidade de tempo do ataque clássico ao gerador Blum-Micali é necessário considerar as operações realizadas e o tamanho da entrada.

A exponenciação modular pode ser implementada de forma linear em relação ao tama-

nho do expoente. Mas, ao considerar instâncias reais, em que p é um primo grande (isto é, $p \approx 2^n$), o número de exponenciações modulares requeridas passa a ser exponencial. Outro procedimento necessário para este ataque é o logaritmo discreto, para o qual não se conhece solução de tempo polinomial na Computação Clássica. Assim, embora seja capaz de recuperar o estado interno do gerador Blum-Micali, este ataque mostra-se ineficiente quando sua complexidade computacional é analisada.

A Computação Quântica explora a possibilidade de utilização da Mecânica Quântica na Ciência da Computação. Se construídos, computadores quânticos poderão prover acelerações sobre diversas operações realizadas por computadores convencionais [Ambainis 2004; Nielsen and Chuang 2005]. Esta afirmação é interessante do ponto de vista de ataques a sistemas criptográficos, pois tais melhorias podem viabilizar a realização de diversos ataques. Neste contexto, a próxima seção apresenta a versão quântica do ataque ao gerador Blum-Micali. A construção desta nova versão visa alcançar um aumento na eficiência por meio da utilização das propriedades da Mecânica Quântica.

4.4 Ataque Quântico ao Gerador de Blum-Micali

O ataque quântico ao gerador de Blum-Micali é baseado na mesma idéia do algoritmo clássico apresentado: bits interceptados pelo adversário são utilizados para confirmar ou refutar hipóteses sobre o estado interno do gerador. As principais mudanças entre estes dois algoritmos referem-se aos diferentes paradigmas computacionais adotados.

O ataque quântico é baseado no *algoritmo quântico de busca*, apresentado na Seção 3.1. O algoritmo quântico de busca realiza a pesquisa por uma solução em uma base de dados desordenada. Este algoritmo é adequado para muitos problemas em que a melhor saída é procurar ingenuamente dentre os elementos da base dados até que uma solução seja encontrada [Grover 1997]. Além deste algoritmo, o ataque faz uso do algoritmo quântico para o logaritmo discreto [Shor 1997].

O circuito que implementa o ataque quântico ao gerador Blum-Micali é composto por dois registradores, mostrados a seguir:

1. **Espaço da Solução**. Contém $n = \lceil \log p \rceil$ qubits que serão utilizados para codificar os elementos em $\mathbb{Z}_p^*$. Todos os qubits deste registrador devem ser inicializados com $|0\rangle$;

2. **Qubits Auxiliares**. Para cada bit do gerador que o adversário interceptou deve corresponder um qubit auxiliar neste registrador. Todos os j qubits auxiliares devem ser inicializados com $|0\rangle$.

De acordo com os procedimentos de inicialização descritos, o estado de entrada do algoritmo é:

$$|\psi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes j} \tag{4.7}$$

O ataque que recupera o representante do estado interno do gerador consiste em duas partes principais: a primeira parte é responsável pela *identificação do representante* $x_j \in X(j)$, e a segunda parte realiza uma *amplificação de amplitude* neste elemento, aumentando a probabilidade do mesmo ser retornado após uma medição. O circuito que implementa esta parte do ataque é ilustrado na Figura 4.4, na qual as duas partes descritas encontram-se enfatizadas.

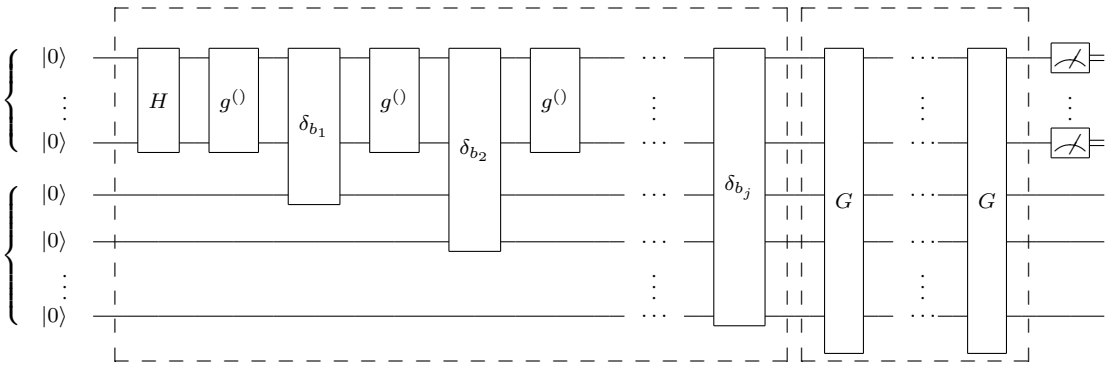

Figura 4.4: Circuito quântico que implementa o ataque que recupera o representante do estado interno de um gerador Blum-Micali.

As duas seções a seguir irão prover detalhes sobre as partes deste algoritmo e das portas quânticas utilizadas no circuito apresentado.

4.4.1 Identificação do Representante

A primeira parte do algoritmo, composta pelas portas H, $g^{()}$ e δ_{b_i} , caracteriza um procedimento para *marcar* o representante. Marcar o representante significa a associação do mesmo ao estado $|1\rangle^{\otimes j}$ no segundo registrador.

A identificação do representante começa com a aplicação da porta de Hadamard ao primeiro registrador. Esta operação resulta em uma superposição igualmente distribuída na

qual todos os elementos do domínio estão representados, i.e., todos os elementos em $\mathbb{Z}_p^*$. O resultado desta operação é denotado em $|\psi_1\rangle$:

$$|\psi_1\rangle = H^{\otimes n} \otimes \mathbb{I}^{\otimes j} |\psi_0\rangle \tag{4.8}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle |0\rangle^{\otimes j} \tag{4.9}$$

As portas $g^{()}$ e δ_{b_i} são definidas em função da permutação *one-way* e do predicado difícil do gerador sob ataque. Elas realizam as seguintes transformações:

$$\delta_{b_i} |x\rangle |y\rangle = \begin{cases} |x\rangle |\overline{y}\rangle, & \text{se } x \in \mathbb{Z}_p^* \text{ e } \delta(x) = b_i \\ |x\rangle |y\rangle, & \text{caso contrário.} \end{cases} \tag{4.10}$$

$$g^{()} |x\rangle = \begin{cases} |g^x \bmod p\rangle, & \text{se } x \in \mathbb{Z}_p^* \\ |x\rangle, & \text{caso contrário.} \end{cases} \tag{4.11}$$

Na definição da porta δ_{b_i}, a operação $|\overline{y}\rangle$ denota a inversão de $|y\rangle$ por meio da aplicação da porta X.

A aplicação das portas $g^{()}$ e δ_{b_i} corresponde a obtenção de um conjunto de estimadores do estado interno do gerador. Os elementos no estado interno do gerador são progressivamente associados a $|1\rangle^{\otimes j}$ no segundo registrador.

O estado após esta primeira parte do algoritmo é dado por:

$$|\psi_2\rangle = \alpha_{x_j} |x_j\rangle |1 \ldots 1\rangle + \sum_{k=0, k\neq x_j}^{2^n-1} \alpha_k |k\rangle |y \neq 1 \ldots 1\rangle \tag{4.12}$$

em que $\left|\alpha_{x_j}\right|^2 + \sum_{w=0, w\neq x_j}^{2^n-1} |\alpha_w|^2 = 1$. Sejam $p_{x_j} = \left|\alpha_{x_j}\right|^2$ e $p_{\neg x_j} = 1 - p_{x_j}$ as probabilidades de obter e de não obter o estado interno do gerador após uma medição, respectivamente. Como mostrado no Lema 4.2.1, o representante está sempre no estado interno do gerador, assim $p_{x_j} > 0$. Uma vez que as amplitudes permanecem inalteradas, pode-se afirmar também que $0 < p_{x_j} < 1$. Neste cenário, as seguintes re-normalizações podem ser utilizadas:

$$\left|\psi_{x_j}\right\rangle = \frac{\alpha_{x_j}}{\sqrt{p_{x_j}}} \left|x_j\right\rangle \left|1 \ldots 1\right\rangle \tag{4.13}$$

$$\left|\psi_{\neg x_j}\right\rangle = \sum_{k=0, k\neq x_j}^{2^n-1} \frac{\alpha_k}{\sqrt{p_{\neg x_j}}} \left|k\right\rangle \left|y \neq 1 \ldots 1\right\rangle \tag{4.14}$$

$$\tag{4.15}$$

Considerando as convenções estabelecidas, o estado $|\psi_2\rangle$ pode ser re-escrito como:

$$|\psi_2\rangle = \sqrt{p_{x_j}} \left|\psi_{x_j}\right\rangle + \sqrt{p_{\neg x_j}} \left|\psi_{\neg x_j}\right\rangle \tag{4.16}$$

Considerando uma representação geométrica:

$$|\psi_2\rangle = \sin(\theta) \left|\psi_{x_j}\right\rangle + \cos(\theta) \left|\psi_{\neg x_j}\right\rangle \tag{4.17}$$

em que $\theta \in \left(0, \frac{\pi}{2}\right)$ satisfaz $\sin^2(\theta) = p_{x_j}$ [Kaye et al. 2007].

Após a conclusão da primeira parte do algoritmo, o representante do estado interno do gerador no tempo j é identificado por meio da associação a $|1\rangle^{\otimes j}$ no segundo registrador. Porém, a amplitude deste elemento possui o mesmo valor dos demais. Isto significa que uma medição no primeiro registrador após a primeira parte do ataque quântico irá retornar qualquer número da superposição com a mesma probabilidade, indicando que o processo de marcação operou exclusivamente no nível quântico. Para trazer esta informação para o nível clássico é necessário amplificar a amplitude do representante, procedimento realizado na segunda parte do algoritmo que será apresentado na próxima seção.

4.4.2 Amplificação de Amplitude

A amplificação de amplitude irá aumentar a probabilidade do representante ser medido em detrimento da probabilidade dos demais elementos. Esta parte do algoritmo é composta por iterações, denotadas pela porta G na Figura 4.4. Cada iteração encapsula a atuação das portas F e A, mostradas em detalhes na Figura 4.5.

A porta F implementa uma inversão de fase no primeiro registrador quando este encontra-se associado a $|1\rangle^{\otimes j}$ no registrador auxiliar. Após esta operação, a porta A amplifica a amplitude daqueles componentes cuja fase encontra-se invertida. A atuação de tais portas no estado de entrada (vide Eq. (4.17)), em termos de iterações, é sumarizada a seguir:

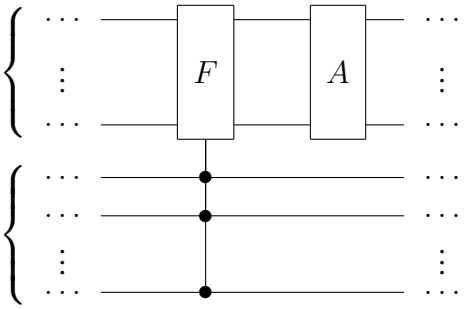

Figura 4.5: Decomposição da porta G, responsável pela amplificação de amplitude.

$$G^k \left|\psi_2\right\rangle = \sin\left((2 \cdot k + 1) \cdot \theta\right) \left|\psi_{x_j}\right\rangle + \cos\left((2 \cdot k + 1) \cdot \theta\right) \left|\psi_{\neg x_j}\right\rangle \tag{4.18}$$

O número k é o número ótimo de iterações requerido para maximizar a amplificação de amplitude do representante.

A determinação do valor de k considera que o adversário interceptou $n = \lceil \log p \rceil$ bits, número requerido para identificar precisamente o estado interno do gerador, como visto na Seção 4.3.2. Assim, o número de iterações k é dado por:

$$k = \left\lfloor \frac{\pi}{4} \sqrt{2^n} \right\rceil \tag{4.19}$$

em que $\lfloor \cdot \rceil$ denota a função inteiro mais próximo.

Após a conclusão da segunda parte do algoritmo, espera-se que, com alta probabilidade, o adversário seja capaz de recuperar o estado interno do gerador. Se portas de medição forem adicionadas ao segundo registrador, o adversário pode também verificar se o resultado da medição em tais portas retorna $1 \ldots 1$, indicando que este elemento é, de fato, o representante. Este procedimento pode evitar algumas repetições do algoritmo para checar a solução correta.

Em posse do representante do estado interno, aplicações do algoritmo quântico para o logaritmo discreto, apresentado na Seção 3.2, devem ser realizadas com o intuito de recuperar todos os elementos do estado interno do gerador, concluindo assim o ataque quântico de comprometimento permanente ao gerador Blum-Micali.

Para exemplificar o ataque quântico descrito, a próxima seção apresenta todos os passos para obtenção do estado interno de um gerador Blum-Micali.

4.4.3 Exemplo do Ataque Quântico ao Gerador Blum-Micali

Para ilustrar o ataque quântico ao gerador de Blum-Micali, considera-se que o adversário conhece os parâmetros públicos $p = 7$ e $g = 3$, e que este descobriu três bits seqüenciais $\mathbf{b}(3) = \{001\}$. De acordo com os procedimentos descritos anteriormente, o circuito para realizar o ataque nestas condições é mostrado na Figura 4.6.

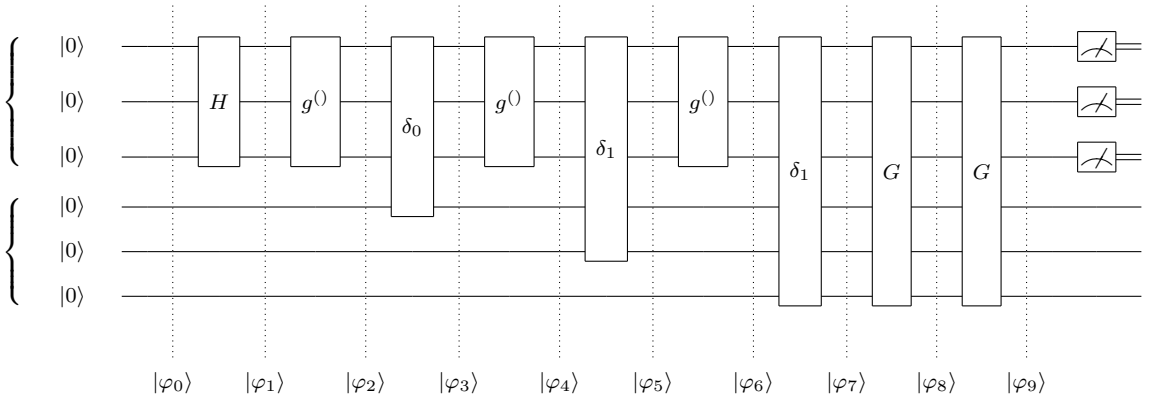

Figura 4.6: Circuito Quântico que exemplifica um ataque ao gerador Blum-Micali.

Os passos que caracterizam o ataque serão descritos de acordo com a evolução do estado $|\varphi\rangle$, de $|\varphi_0\rangle$ até $|\varphi_9\rangle$. A entrada do algoritmo é preparada no seguinte estado, de acordo com as regras de definição de cada registrador:

$$|\varphi_0\rangle = |000\rangle\,|000\rangle\,. \tag{4.20}$$

O próximo passo do algoritmo é a aplicação da porta H ao primeiro registrador, resultando:

$$|\varphi_1\rangle = \left[\frac{1}{\sqrt{8}}\left(|0\rangle + |1\rangle + \ldots + |7\rangle\right)\right]|000\rangle\,. \tag{4.21}$$

É possível visualizar que todos os elementos do domínio $\{1, 2, \ldots, 6\}$ encontra-se representados no estado $|\varphi_1\rangle$. O próximo passo do algoritmo consiste na aplicação da porta $g^{()}$, que possui a seguinte definição:

$$g^{()} = |0\rangle\langle 0| + |3\rangle\langle 1| + |2\rangle\langle 2| + |6\rangle\langle 3| + |4\rangle\langle 4| + |5\rangle\langle 5| + |1\rangle\langle 6| + |7\rangle\langle 7| \tag{4.22}$$

O estado $|\varphi_2\rangle$, resultante da aplicação de $g^{()}$ é:

$$\begin{aligned} |\varphi_2\rangle &= \frac{1}{\sqrt{8}}\left(|0\rangle|000\rangle + |3\rangle|000\rangle + |2\rangle|000\rangle + |6\rangle|000\rangle + \right. \\ &+ \left. |4\rangle|000\rangle + |5\rangle|000\rangle + |1\rangle|000\rangle + |7\rangle|000\rangle\right). \end{aligned} \tag{4.23}$$

A aplicação da porta δ_0 irá associar a $|1\rangle$ no primeiro registrador do segundo qubit todos os elementos em $\mathbb{Z}_p^*$ que poderiam ter produzido o bit $b_1 = 0$. O resultado de tal operação é dada por:

$$\begin{aligned} |\varphi_3\rangle &= \frac{1}{\sqrt{8}}\left(|0\rangle|000\rangle + |3\rangle|100\rangle + |2\rangle|100\rangle + |6\rangle|000\rangle + \right. \\ &+ \left. |4\rangle|000\rangle + |5\rangle|000\rangle + |1\rangle|100\rangle + |7\rangle|000\rangle\right). \end{aligned} \tag{4.24}$$

Resumindo a aplicação das quatro portas quânticas seguintes, o estado resultante após a identificação do representante é dado por:

$$\begin{aligned} |\varphi_7\rangle &= \frac{1}{\sqrt{8}}\left(|0\rangle|000\rangle + |1\rangle|100\rangle + |2\rangle|110\rangle + |3\rangle|010\rangle + \right. \\ &+ \left. |4\rangle|001\rangle + |5\rangle|001\rangle + |6\rangle|111\rangle + |7\rangle|000\rangle\right) \end{aligned} \tag{4.25}$$

É importante observar que o único estado associado a 111 no segundo registrador é $|6\rangle$. É também relevante enfatizar que todas as amplitudes são iguais, i.e., uma medição retornaria qualquer número de 0 a 7 com a mesma probabilidade.

Re-escrevendo o estado $|\varphi_7\rangle$ como uma partição:

$$\begin{aligned} |\varphi_7\rangle &= \frac{1}{\sqrt{8}}|6\rangle|111\rangle + \frac{1}{\sqrt{8}}\left(|0\rangle|000\rangle + |1\rangle|100\rangle + |2\rangle|110\rangle + \right. \\ &+ \left. |3\rangle|010\rangle + |4\rangle|001\rangle + |5\rangle|001\rangle + |7\rangle|000\rangle\right) \\ &= \frac{1}{\sqrt{8}}|x_j\rangle|111\rangle + \sqrt{\frac{7}{8}}|\neg x_j\rangle|y\rangle \\ &= \sin(\theta)\left|\psi_{x_j}\right\rangle + \cos(\theta)\left|\psi_{\neg x_j}\right\rangle \end{aligned} \tag{4.26}$$

em que $j = 3$; $|x_j\rangle = |6\rangle$; $\neg x_j$ denota todos os elementos que não são o representante x_j; $y \neq 111$; $\left|\psi_{x_j}\right\rangle = |6\rangle|111\rangle$; $\left|\psi_{\neg x_j}\right\rangle = |\neg x_j\rangle|y\rangle$; e, $\theta \in \left(0, \frac{\pi}{2}\right)$ satisfaz $\theta = \arcsin\left(\frac{1}{\sqrt{8}}\right) = 0.36$ radianos.

O próximo passo do algoritmo consiste em uma amplificação de amplitude. Antes deste passo, é necessário determinar quantas iterações de Grover são necessárias. Levando em consideração que $n = 3$, então $k = \left\lfloor \frac{\pi}{4}\sqrt{2^3} \right\rfloor = 2$ iterações.

Assim, duas iterações de Grover no estado $|\varphi_7\rangle$ irão resultar em:

$$\begin{aligned} |\varphi_9\rangle &= G^2 |\varphi_7\rangle && (4.27) \\ &= \sin((2\cdot 2+1)\cdot\theta)\left|\psi_{x_j}\right\rangle + \cos((2\cdot 2+1)\cdot\theta)\left|\psi_{\neg x_j}\right\rangle \\ &= \sin(5\theta)\left|\psi_{x_j}\right\rangle + \cos(5\theta)\left|\psi_{\neg x_j}\right\rangle \\ &= \sin(1.8)\left|\psi_{x_j}\right\rangle + \cos(1.8)\left|\psi_{\neg x_j}\right\rangle && (4.28) \end{aligned}$$

Uma medição em $|\varphi_9\rangle$ irá resultar em 6 com uma probabilidade de $|\sin(1.8)|^2 \approx 94.83\%$. Considerando que o adversário recuperou o representante do estado interno do gerador após uma medição, ele deve proceder com duas aplicações do algoritmo quântico para o logaritmo discreto, resultando em $\log_3 6 = 3$ e $\log_3 3 = 1$. Isto implica que o adversário recuperou $X(3) = \{6, 3, 1\}$ e, com esta informação, pôde concluir com sucesso o ataque quântico de comprometimento permanente ao gerador Blum-Micali.

4.5 Análise do Ataque Quântico ao Gerador de Blum-Micali

Nesta seção, o ataque quântico para o gerador Blum-Micali é discutido em detalhes. Esta discussão contempla a apresentação da análise de complexidade, na qual os custos do algoritmo são determinados e comparados com o caso clássico, uma argumentação acerca da impossibilidade de uso do algoritmo quântico para o logaritmo discreto para a recuperação do representante e, por fim, considerações sobre a construção das portas requeridas pelo circuito.

4.5.1 Análise da Complexidade

Para realizar a análise de complexidade de tempo deste algoritmo, dois momentos devem ser considerados: a obtenção do representante do estado interno do gerador e a obtenção de todo

o estado interno.

Com respeito ao primeiro momento, a identificação do representante utiliza $2 \cdot j + 1$ portas com complexidade $O(1)$ cada. Considerando que $j = \log p$ e $p \approx 2^n$, a complexidade resultante desta primeira fase é $O(n)$, ou seja, linear em relação a entrada. A amplificação de amplitude requer $\sqrt{2^n}$ operações com custo unitário cada. Assim, a complexidade resultante do primeiro momento é $O(n)+O(\sqrt{2^n}) = O(\sqrt{2^n})$, representando um ganho *quadrático* em relação em relação a sua contrapartida clássica. Este resultado reproduz o ganho alcançando quando o algoritmo de Grover é utilizado na solução de problemas $\mathcal{NP}$-completos [Nielsen and Chuang 2005].

No segundo momento, há um ganho significativo quando o paradigma da Computação Quântica é utilizado. Isto é verificado porque não há nenhum algoritmo de tempo polinomial conhecido na Computação Clássica para realizar logaritmos discretos de maneira eficiente, enquanto que na Computação Quântica há o algoritmo proposto por Shor [1997]. Em virtude disto, este segundo momento atinge um ganho *superpolinomial* em relação ao algoritmo clássico equivalente.

Em relação a complexidade do circuito quântico adotado, três métricas são possíveis de serem determinadas. A primeira delas, o número total de portas do circuito, é igual a $2 \cdot j + \sqrt{2^n} + 1 + n$ portas; a segunda delas, a profundidade do circuito é igual a $2 \cdot j + \sqrt{2^n} + 2$ passos; e, por fim, a largura do circuito, é igual a $n + j$ qubits.

4.5.2 Considerações Sobre a Impossibilidade de Uso do Algoritmo Quântico do Logaritmo Discreto para Recuperação do Representante

Embora a segurança do gerador de Blum-Micali seja baseada na hipótese de intratabilidade do problema do logaritmo discreto, o algoritmo quântico de tempo polinomial para este problema não pode ser aplicado diretamente na recuperação do representante do estado interno do gerador.

Para justificar a afirmação anterior, é importante lembrar que o algoritmo em questão demanda dois parâmetros g e $g^{x_i} \bmod p$ que serão utilizados para construir o oráculo que irá retornar x_i, como explicado anteriormente na Seção 3.2.2. Porém, é importante lembrar que

$x_{i+1} = g^{x_i} \bmod p$ e x_{i+1} é um elemento do estado interno do gerador.

Para satisfazer o requisito de entrada do algoritmo quântico do logaritmo discreto, que demanda um elemento do estado interno do gerador *a priori*, é necessário quebrar a premissa de segurança inicialmente considerada, que afirma que os elementos do estado interno do gerador devem ser secretos. Porém, quebrar a premissa de segurança já torna o ataque trivial. Diante disto, não é possível definir um procedimento para recuperação do representante com o algoritmo quântico para o logaritmo discreto.

4.5.3 Construção das Portas Requeridas

Para que o algoritmo quântico descrito possa ser implementado fisicamente, é necessário analisar a viabilidade da construção das portas utilizadas pelo mesmo.

Em essência, as portas do tipo δ_{b_i} realizam comparações do tipo *maior que* e *menor que*, armazenando o resultado em um qubit auxiliar. Este tipo de operação é implementado eficientemente em nível clássico e, por extensão, também em nível quântico [Bennett 1973]. Em termos de portas universais, a porta δ_{b_i} demanda apenas portas do tipo $CNOT$ para ser implementada, como pode ser visto na Figura 4.7 referente a uma porta do tipo δ_1 utilizada no exemplo da Seção 4.4.3. É importante mencionar que nesta figura considera-se que o primeiro qubit é o menos significativo e que o último qubit é o qubit auxiliar (alvo) correspondente no segundo registrador.

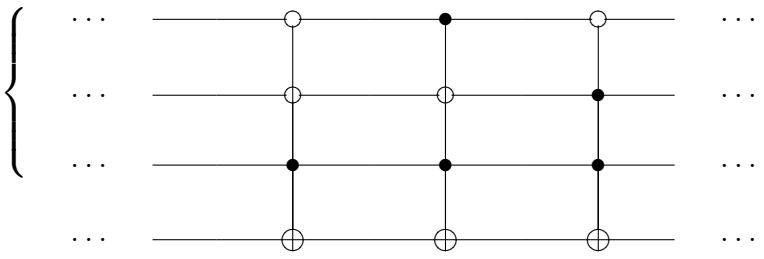

Figura 4.7: Exemplo de decomposição da porta δ_{b_1} em portas CNOT.

A porta $g^{()}$ implementa uma permutação. De acordo com Williams [2011], uma permutação quântica pode ser implementada a partir de portas R_z, R_y, fase e $CNOT$ sem utilização de qubits auxiliares, ou então a partir de portas X, $CNOT$ e Toffoli demandando, no máximo, um qubit auxiliar a depender do número de permutações realizadas. As portas R_z e R_y correspondem as matrizes de Pauli Z e Y quando rotacionadas sobre os eixos $\hat{z}$ e $\hat{y}$,

respectivamente.

Não é possível determinar diretamente a quantidade de portas elementares necessárias para implementar as operações δ_{b_i} e $g^{()}$. Isto acontece porque a definição de tais portas está vinculada ao gerador sob ataque e não há um caso geral que as descreva. Além disso, não há procedimentos computacionais automatizados para esta tarefa. Detalhes em particular sobre este aspecto serão discutidos na Seção 6.2.

A construção das portas F e A, utilizadas na etapa de amplificação de amplitude, segue as recomendações de Grover [1997], que argumenta que tais portas baseiam-se apenas em transformações de Hadamard e deslocamentos de fase condicionais, operações estas que são simples de serem implementadas quando comparadas a procedimentos demandados por outros algoritmos quânticos, a exemplo da Transformada de Fourier.

Notas do Capítulo

Este capítulo apresentou inicialmente o gerador de Blum-Micali, um CSPRNG baseado na hipótese de intratabilidade do problema do logaritmo discreto. Após a apresentação deste gerador, um ataque clássico para o mesmo foi delineado. Este ataque clássico baseia-se na existência de um adversário e na interceptação de bits da saída do gerador, sendo classificado como um ataque de comprometimento permanente. A complexidade computacional deste algoritmo foi discutida, assim como o número de bits requeridos para realizar o ataque com sucesso.

Na tentativa de melhorar a eficiência do ataque clássico apresentado, um ataque quântico foi proposto, baseado no algoritmo quântico de busca. Este ataque quântico é composto de duas partes: na primeira há a identificação do representante e, em seguida, há a amplificação de amplitude do mesmo, aumentando a probabilidade deste ser retornado após uma medição. Depois desta etapa, o adversário pode utilizar o algoritmo quântico para o logaritmo discreto com o intuito de obter todo o estado interno do gerador.

O ataque quântico apresentado possui ganhos significativos em relação à sua contrapartida clássica: um ganho quadrático em um primeiro momento, seguido de um ganho superpolinomial. Este ataque demonstra uma maior vulnerabilidade do gerador Blum-Micali a ataques realizados pelo paradigma da Computação Quântica.

Capítulo 5

Generalizações do Ataque Quântico

O ataque quântico apresentado no capítulo anterior visa a quebra de imprevisibilidade do gerador Blum-Micali. Se algumas modificações forem realizadas, este ataque pode ser generalizado, a ponto de ser visto como um *framework* de ataques para certos geradores pseudo-aleatórios. Deste modo, tais generalizações ampliam o escopo das respostas para as questões motivadoras deste trabalho.

A primeira generalização deste ataque torna-o capaz de ameaçar geradores de Blum-Micali com múltiplos predicados difíceis. A segunda generalização amplia o ataque para outros geradores da construção de Blum-Micali. As duas generalizações seguintes adequam o ataque proposto a situações mais realísticas: quando o adversário não possui bits consecutivos e quando a quantidade de bits é inferior ao requerido. Tais generalizações serão apresentadas em detalhes nas seções a seguir.

É importante mencionar que as generalizações aqui apresentadas podem ser usadas de forma ampla, por exemplo, para atacar um gerador Blum-Blum-Shub com múltiplos predicados, ou para atacar um gerador Kaliski com poucos bits e não consecutivos.

5.1 Geradores com Múltiplos Predicados

Embora o gerador de Blum-Micali seja adotado em diversos sistemas criptográficos, este gerador requer um certo esforço computacional para produzir bits. De acordo com a definição original, apresentada na Seção 4.1, apenas um bit é extraído por iteração, i.e., por exponenciação modular realizada.

Na tentativa de melhorar a produção de bits sem ameaçar a segurança deste gerador, outros autores exploraram a possibilidade de extrair mais bits por iteração. Long e Wigderson [1988] e Peralta [1986] mostraram que é possível extrair $O(\log \log p)$ bits em uma única iteração do gerador Blum-Micali sem comprometer a segurança do mesmo. Posteriormente, Håstad et al. [1993] mostraram que, ao realizar o logaritmo discreto de um número composto, até $n/2$ bits podem ser extraídos por iteração. Outros autores mostraram que extrações similares são possíveis mesmo quando o expoente é um número pequeno [Patel and Sundaram 1998; Gennaro 2005]. Em síntese, os trabalhos mencionados tentam prover mais eficiência na produção de bits do gerador Blum-Micali por meio da adição de múltiplos predicados difíceis.

Para cada predicado difícil γ de um gerador de Blum-Micali existe uma porta quântica associada. Tal porta, diga-se $\mathcal{Y}$, é capaz de determinar se um certo elemento do registrador de controle x produziu um bit b ($\gamma(x) = b$) por meio do registro desta informação em um qubit alvo y, de acordo com o seguinte procedimento:

$$\mathcal{Y}_b \left|x\right\rangle \left|y\right\rangle = \begin{cases} \left|x\right\rangle \left|\overline{y}\right\rangle, & \text{se } x \in \mathbb{Z}_p^* \text{ e } \gamma(x) = b \\ \left|x\right\rangle \left|y\right\rangle, & \text{caso contrário.} \end{cases} \tag{5.1}$$

É importante ressaltar que a definição da porta $\mathcal{Y}_b$ é reversível. Basta verificar que, para implementá-la, é necessário utilizar apenas portas do tipo $CNOT$, como descrito na Seção 4.5.3.

Se um adversário almeja atacar uma versão do gerador de Blum-Micali com múltiplos predicados, basta armazenar em **b** as tuplas de bits interceptados a cada iteração, mantendo a ordem e o predicado de origem dos mesmos. Por exemplo, suponha que um gerador de Blum-Micali possui três predicados difíceis. Ao interceptar bits deste gerador, o adversário verifica que cada um dos predicados difíceis produziu os bits 111, 011 e 100, respectivamente. De acordo com os procedimentos descritos, este adversário deve organizar em **b** as seguintes tuplas: $\{(1,0,1),(1,1,0),(1,1,0)\}$.

Na construção do circuito para o ataque quântico, o adversário deve considerar um registrador extra para cada predicado difícil. A dimensão deste registrador deve ser igual a quantidade de bits interceptados do predicado correspondente, respeitando as especificações de inicialização descritas na Seção 4.4. As portas relativas aos predicados difíceis devem ser

inseridas entre duas portas $g^{()}$, que implementam as exponenciações modulares do gerador.

O procedimento descrito permite o uso do ataque em situações mais gerais. Isto é uma extensão importante, especialmente quando se leva em consideração que múltiplos predicados em um gerador Blum-Micali produzem mais bits por iteração e, por conseguinte, é uma melhoria nestes geradores mais provável de ser utilizada em sistemas criptográficos do mundo real. Além disso, tal extensão permite atacar o gerador Blum-Micali de forma mais efetiva, pois mais informação é capturada do gerador em cada iteração.

5.2 Construção de Blum-Micali

A *construção de Blum-Micali* é uma família de geradores pseudo-aleatórios definidos a partir de uma permutação *one-way* f sobre um domínio $\mathcal{D}$, e de um predicado difícil ϕ para f [Sidorenko 2008]. A semente é escolhida aleatoriamente no domínio, i.e., $x_0 \in_R \mathcal{D}$. As produções (bits) destes geradores são obtidos da seguinte forma:

$$x_i = f(x_{i-1}) \tag{5.2}$$

$$b_i = \phi(x_i). \tag{5.3}$$

Três geradores são os principais representantes desta família: o gerador Blum-Micali, apresentado na Seção 4.1, o gerador Blum-Blum-Shub e o gerador de Kaliski.

A permutação *one-way* do gerador Blum-Blum-Shub é a função de Rabin $f : \mathbb{Z}_M \to \mathbb{Z}_M$, tal que $f(x) = x^2 \bmod M$, em que M é o produto de dois números primos congruentes a $3 \bmod 4$. O domínio deste gerador é $QR_M = (\mathbb{Z}_M^*)^2$. O predicado difícil deste gerador retorna o j-ésimo bit do parâmetro de entrada, em que j é previamente fixado. Os valores M e j são divulgados publicamente. O gerador Blum-Blum-Shub é considerado um dos geradores mais eficientes conhecidos e baseia-se na hipótese de intratabilidade da fatoração de inteiros grandes [Blum et al. 1986a].

O gerador Kaliski é baseado em curvas elípticas do logaritmo discreto. Sejam p um primo, $p \equiv 2 \bmod 3$, e a curva $E(\mathbb{F}_p)$ dos pontos $(x, y) \in \mathbb{F}_p \times \mathbb{F}_p$ tais que $y^2 = x^3 + c$, em que $c \in \mathbb{F}_p$. Os pontos $E(\mathbb{F}_p)$ juntamente com um ponto no infinito, denotado por $\mathcal{O}$, formam um grupo cíclico aditivo de ordem $p + 1$, que é o domínio do gerador. Sejam Q um

gerador deste grupo e φ uma função com a seguinte definição:

$$\varphi(P) = \begin{cases} y, & \text{se } P \in E(\mathbb{F}_p) \\ p, & \text{se } P = \mathcal{O}. \end{cases} \tag{5.4}$$

A semente do gerador Kaliski é um ponto sob a curva escolhido de forma aleatória. A permutação *one-way* é a função $f(P) = \varphi(P) \cdot Q$ e o predicado difícil deste gerador retorna 1 se $\varphi(P) \geq \frac{p+1}{2}$, e 0 em caso contrário. A segurança deste gerador baseia-se na hipótese de que o problema das curvas elípticas do logaritmo discreto é intratável [Kaliski 1988].

Considerando que cada permutação *one-way* define uma bijeção sobre o seu domínio, é possível construir portas quânticas capazes de implementar tais funções [Williams 2011]. Seja f uma permutação *one-way* sobre o domínio $\mathcal{D}$, e seja $\mathcal{Q}$ uma porta quântica que implementa esta operação. A porta $\mathcal{Q}$ realiza as seguintes transformações:

$$\mathcal{Q}\left|x\right\rangle = \begin{cases} \left|f(x)\right\rangle, & \text{se } x \in \mathcal{D} \\ \left|x\right\rangle, & \text{caso contrário.} \end{cases} \tag{5.5}$$

Como visto na Eq. (5.1), há uma porta quântica associada a cada predicado difícil. Isto torna possível a construção de ataques análogos ao ataque descrito na Seção 4.4. A principal modificação diz respeito a parte de identificação do representante, em que as portas $g^{()}$ e δ_{b_i} devem ser substituídas pelas portas equivalentes do gerador da construção de Blum-Micali sob ataque.

O ganho alcançado quando a Computação Quântica é utilizada para atacar os demais geradores da construção de Blum-Micali é o mesmo alcançado para atacar o gerador de Blum-Micali, i.e., um ganho quadrático relativo à recuperação do representante do estado interno do gerador seguido de um ganho superpolinomial para a recuperação de todo o estado interno. Este último ganho só é possível devido a existência de algoritmos quânticos de tempo polinomial para o logaritmo discreto, fatoração e curvas elípticas do logaritmo discreto [Shor 1997; Proos and Zalka 2004].

Exemplos detalhados da instanciação e da elaboração de ataques aos geradores da construção de Blum-Micali podem ser vistos no artigo de Guedes et al. [2010a]. A realização de ataques a geradores da construção de Blum-Micali com múltiplos predicados é permitida e deve seguir os procedimentos descritos na Seção 5.1.

5.3 Ataques com Bits Não-Consecutivos

O ataque quântico ao gerador Blum-Micali e suas generalizações, como definidos, não comportam uma situação na qual o adversário pode capturar uma seqüência de bits não-consecutivos. Porém, ainda assim é possível adaptar o ataque a esta situação bastante realística.

Para realizar o ataque nestas condições, o adversário deve manter em **b** os bits capturados e a posição a que correspondem. Por exemplo, se o adversário capturou os três primeiros bits, o quinto e o sétimo, deve armazená-los em **b** da seguinte maneira: $\mathbf{b} = \{b_1 = 1, b_2 = 0, b_3 = 1, b_5 = 0, b_7 = 1\}$.

O próximo passo para elaborar o ataque é montar o circuito correspondente. Para tanto, deve-se levar em consideração que se há bits consecutivos, a montagem do ataque é feita de forma convencional, como descrito na Seção 4.4.1. Para o caso de bits não consecutivos, as portas δ correspondentes devem ser separadas por tantas portas $g^{()}$ quanto for o módulo da diferença entre os índices desses bits. Por exemplo, considerando que o gerador sob ataque é o Blum-Micali, duas aplicações da porta $g^{()}$ devem ser realizadas entre as portas δ_{b_5} e δ_{b_7}, pois $|7 - 5| = 2$.

A prova de que esta adaptação ainda recupera o estado interno é apresentada a seguir. Ela baseia-se no fato de que, por não haver bits sucessivos em dado momento, elementos não são descartados dos conjuntos de estimadores. Em decorrência, os elementos passam apenas por mapeamentos, que reproduzem o comportamento do gerador sob ataque.

Lema 5.3.1. *Dado um conjunto de estimadores $\hat{X}_i$, então existe x_k tal que $k - i > 1$, $x_k \in X(k)$ e $x_k \in \hat{X}_k$.*

Prova. *Do Lema 4.2.1, sabe-se que $x_i \in X(i)$ e $x_i \in \hat{X}_i$. Nesta situação, uma vez que não há bits disponíveis, então $\left|\hat{X}_i\right| = \left|\hat{X}_k\right|$. Como $x_i \in \hat{X}_i$, então existe $x_k \in \hat{X}_k$ tal que $x_i \mapsto \ldots \mapsto x_k$. Dado que $x_i \in X(k)$, segue que existe $x_k \in X(k)$.* □

Para o exemplo em questão, em que o adversário possui os bits $\mathbf{b} = \{b_1 = 1, b_2 = 0, b_3 = 1, b_5 = 0, b_7 = 1\}$, o circuito da parte de identificação do representante que realiza o ataque é mostrado na Figura 5.1.

As especificações do número de iterações necessárias para amplificação de amplitude e do número de bits requeridos segue a mesma idéia que o ataque ao gerador de Blum-Micali

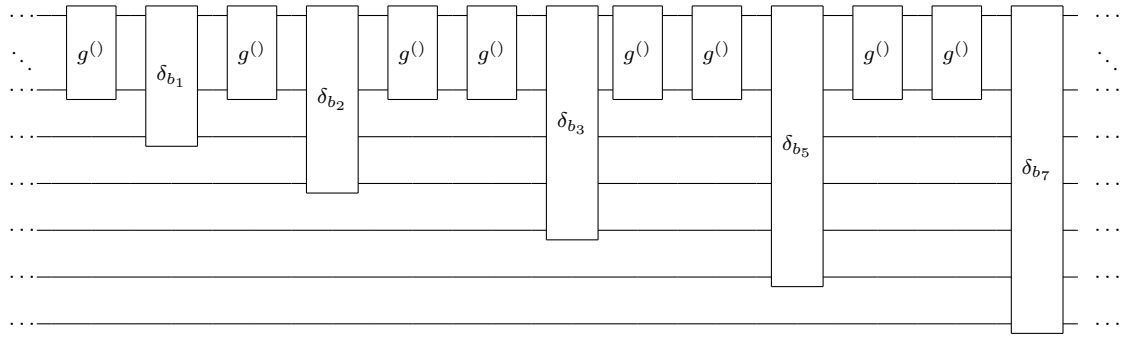

Figura 5.1: Exemplo de utilização das portas δ_{b_i} e $g^{()}$ na parte de identificação do representante quando os bits não são consecutivos.

com bits consecutivos. A principal diferença entre estas duas situações é que, quando os bits não são consecutivos, há um número maior de portas quânticas utilizadas.

Este mesmo procedimento pode ser aplicado a geradores da construção de Blum-Micali com um ou mais predicados difíceis, respeitando as devidas adaptações para estas situações.

5.4 Ataques com Menos Bits que o Requerido

Outra situação bastante realística é a realização de ataques quando a quantidade de bits requerida é menor que $\log p$. Implementar ataques neste cenário pode trazer vantagens ao adversário, mas o sucesso é menos provável, pois a quantidade de bits inferior ao requerido impede a geração de conclusões sobre o estado interno do gerador com alto índice de certeza.

Como esclarecido na Seção 4.3.2, cada bit interceptado pelo adversário corresponde a um bit do representante do estado interno do gerador. Se o adversário interceptou x bits, tal que $x < \log p$, então há $(\log p - x) \cdot 2$ números possíveis que podem ser o representante do estado interno do gerador, com a mesma probabilidade. Neste cenário, o número de bits interceptados não determina com precisão quem é o representante, mas elege um conjunto de elementos mais prováveis e retorna um destes após a medição.

A montagem do circuito de ataque quântico nestas condições segue os mesmos procedimentos descritos nas Seção 4.4. A principal mudança diz respeito ao número de iterações a serem realizadas para amplificação de amplitude, que segue o procedimento para o algoritmo quântico de busca com múltiplas soluções [Chen et al. 2001]. Neste caso, o valor de k é calculado da seguinte forma:

$$k = \left\lfloor \frac{\pi}{4} \sqrt{\frac{2^n}{(\log p - x) \cdot 2}} \right\rceil \tag{5.6}$$

em que $\lfloor \cdot \rceil$ denota a função inteiro mais próximo. No caso de geradores da construção de Blum-Micali, o valor $(\log p - x) \cdot 2$ deve ser substituído por $(\log |\mathcal{D}| - x) \cdot 2$.

Embora não possa determinar o representante de forma precisa, realizar o ataque com poucos bits pode favorecer o adversário, pois existe uma probabilidade de ainda assim obter o elemento desejado. O ganho ao utilizar Computação Quântica nesta situação é o mesmo do cenário em que o adversário conhece a quantidade de bits adequada, porém tais situações diferenciam-se pela probabilidade de encontrar a solução desejada.

Notas do Capítulo

Neste capítulo foram apresentadas generalizações do ataque quântico previamente proposto. Estas generalizações visam uma maior abrangência do ataque, possibilitando a adequação do mesmo a um número muito maior de geradores.

A primeira generalização considerou o ataque a geradores de Blum-Micali com múltiplos predicados, ou seja, capazes de produzir mais de um bit por iteração. Este ataque envolve a utilização de diferentes portas para os predicados utilizados e é mais efetivo, por capturar mais informação por iteração. A segunda generalização amplia o ataque para a construção de Blum-Micali, incluindo os geradores de Blum-Blum-Shub e Kaliski. Esta generalização demanda a definição de portas a partir das permutações e predicados de cada gerador. Os ganhos ao realizar o ataque quântico são os mesmos verificados quando se usa Computação Quântica para atacar o gerador de Blum-Micali. Os dois ataques seguintes envolvem adaptações para situações em que há bits não consecutivos ou menos bits que o requerido. Estas adaptações permitem tirar o maior proveito possível, ainda que o adversário não possua os requisitos ideais para o ataque.

Em todos as generalizações, os ganhos sob a Computação Clássica seguem o que foi verificado no Capítulo 4: um ganho quadrático para obter o representante do estado interno, seguido de um ganho superpolinomial na obtenção dos demais elementos do estado interno. Também foi visto que as generalizações podem ser usadas de forma ampla, ou seja, podem ser combinadas para a realização de um determinado ataque.

Capítulo 6

Considerações Finais

Este capítulo contempla as considerações finais desta dissertação e apresenta algumas sugestões de trabalhos futuros que visam contribuir com a solução apresentada neste documento.

6.1 Conclusões e Contribuições

Este trabalho visou analisar a vulnerabilidade de geradores pseudo-aleatórios contra ameaças da Computação Quântica. Como resultado, foi apresentado um algoritmo quântico de ataque ao gerador Blum-Micali capaz de recuperar o estado interno e, em decorrência, reproduzir seqüências do mesmo, ou seja, capaz de destruir a imprevisibilidade deste gerador. Para realizar este ataque quântico, um adversário precisa conhecer os parâmetros públicos do gerador BM e também interceptar alguns bits da saída deste gerador. O ataque quântico consiste em duas partes: a primeira, denominada identificação do representante, marca, em nível quântico, o representante do estado interno do gerador. A segunda etapa, denominada amplificação de amplitude, aumenta a probabilidade deste representante ser retornado após uma medição. Para que este algoritmo retorne o representante do estado interno com alta probabilidade $\log p$ bits devem ser interceptados.

O ataque quântico proposto utiliza o algoritmo quântico de busca e também demanda o algoritmo quântico para o logaritmo discreto, caso o adversário tenha interesse em remontar todo o estado interno do gerador. Este ataque promove ganhos em relação ao paradigma da Computação Clássica: ganho quadrático em se tratando da obtenção do representante, e ganho superpolinomial relativo a recuperação de todo o estado interno. A existência de

tais ganhos responde a segunda questão de pesquisa, afirmando que é possível realizar tais ataques com mais eficiência utilizando o paradigma quântico que o clássico.

Se algumas modificações forem realizadas, é possível adaptar o ataque, tornando-o uma espécie de *framework*. Estas modificações caracterizam generalizações do ataque proposto e o tornam apto a: atacar geradores de Blum-Micali com múltiplos predicados; atacar geradores da Construção de Blum-Micali, a exemplo do gerador Blum-Blum-Shub e Kaliski; a realizar ataques quando os bits interceptados não são consecutivos; e, ainda, quando há menos bits que o requerido para identificar o representante com alta probabilidade. Estas generalizações podem ser combinadas de forma mais ampla, por exemplo, para atacar um gerador Kaliski com múltiplos predicados.

Em relação a outros ataques ao gerador Blum-Micali, a maioria dos relatos concentra-se na elaboração de ataques com Computação Clássica [Sidorenko and Schoenmakers 2005; Yao 1982]. Apenas o trabalho de Oliveira [2007] é seminal ao propor a utilização do algoritmo quântico de busca na elaboração de ataques a geradores pseudo-aleatórios. Embora proponha um algoritmo para ataque, carecem provas de corretude e análises de reversibilidade das portas adotadas. A adoção do algoritmo quântico de busca também não é feita de forma explícita, o autor indica onde tal procedimento deve ser aplicado e quais os resultados desta aplicação, sem fornecer detalhes sobre a função que o oráculo implementa, por exemplo. Embora o presente trabalho proponha um ataque análogo, há alguns aspectos diferenciais em relação ao trabalho de Oliveira: provas de corretude são apresentadas; a complexidade da largura do circuito quântico é menor, isto é, são usados menos qubits como entrada; o algoritmo quântico de busca para múltiplas soluções é utilizado; e, por fim, a reversibilidade e a complexidade de construção das portas são analisadas. O ataque proposto neste trabalho também se mostra mais amplo, uma vez que pode ser adaptado a outros geradores.

Um dos aspectos inovadores deste trabalho é a determinação do número de bits para atacar o gerador de Blum-Micali. Os resultados obtidos estão de acordo com Boyar [1989] e Krawczyk [1992], que afirmam que, sob certas condições, se o período de um gerador é p, então é possível predizer a saída do mesmo com um número de bits limitado polinomialmente por $\log p$.

Embora o algoritmo quântico de busca seja um dos alicerces do ataque proposto, algu-

mas diferenças entre a definição original do algoritmo podem ser observadas. A primeira delas é que, na definição original, os elementos da base de dados são estáticos, enquanto no ataque quântico tais elementos podem ser marcados e passar por transformações. A segunda modificação é que um registrador auxiliar é necessário para realizar a identificação da solução, cuja amplitude será posteriormente amplificada. A terceira modificação diz respeito a não necessidade de um qubit para inversão de fase do oráculo da busca quântica. Por fim, o oráculo implementa uma inversão de fase controlada. Neste último ponto, em particular, o ataque quântico elimina o uso de uma função caixa-preta.

O trabalho realizado também contribui para uma melhor utilização do gerador Blum-Micali. Ao sugerir a representação do mesmo segundo grafos funcionais, tomando como inspiração o trabalho de Cloutier e Holden [2010], procedimentos de inicialização e escolha da semente são propostos, diminuindo a vulnerabilidade deste gerador contra ataques e aumentando a eficiência do mesmo na produção de seqüências pseudo-aleatórias.

Algumas limitações da solução proposta também são pontuadas. A primeira delas diz respeito a necessidade de interceptar bits para realizar o ataque. Se este requerimento não for atendido, não há como diminuir a incerteza sobre o estado interno do gerador. Este requisito pode inviabilizar o ataque a determinadas arquiteturas de sistemas criptográficos. Outro aspecto em que a solução proposta mostra-se limitada diz respeito ao ganho quadrático em relação a Computação Clássica no tocante a recuperação do representante do estado interno do gerador. Embora este seja um ganho significativo, é importante lembrar que quando p é muito grande ainda há um esforço computacional considerável para realizar o ataque. Este esforço é advindo majoritariamente da complexidade computacional da etapa de amplificação de amplitude.

Sumarizando as considerações realizadas, o trabalho realizado traz contribuições para as áreas de Criptoanálise e Computação Quântica. Em relação a primeira, o trabalho de dissertação: (i) melhora o conhecimento acerca dos possíveis ataques aos PRNGs e CSPRNGs utilizados em aplicações criptográficas do mundo real; (ii) caracteriza algoritmos de ataque a geradores; (iii) analisa a segurança deste componente de diversos sistemas criptográficos contra ameaças da Computação Quântica; e (iv) aponta novos requisitos de segurança para alguns CSPRNGs. No tocante à Computação Quântica, as contribuições alcançadas são: (i) a proposição de algoritmos de acordo com este paradigma; (ii) a caracterização de novas

ameaças à sistemas de criptografia clássicos; e (iii) a verificação da existência de ganhos em relação aos algoritmos clássicos que implementam ataques a geradores.

Como resultado deste trabalho de dissertação, alguns trabalhos foram produzidos:

1. [Guedes et al. 2010c]: Artigo completo em língua inglesa intitulado "*Quantum Permanent Compromise Attack to Blum-Micali Pseudorandom Generator*", publicado no *International Telecommunications Symposium* 2010. Este artigo descreve o ataque quântico ao gerador de Blum-Micali;

2. [Guedes et al. 2010b]: Artigo completo em língua inglesa intitulado "*A Generalized Quantum Permanent Compromise Attack to the Blum-Micali Construction*", publicado no III Workshop-Escola de Computação e Informação Quânticas (WECIQ). Este artigo apresenta a generalização do ataque para a Construção de Blum-Micali e relata os resultados experimentais obtidos sobre o número de bits requeridos para o ataque;

3. [Guedes et al. 2011]: Relatório técnico intitulado "Busca Quântica". Este relatório técnico foi resultado do levantamento bibliográfico sobre o referido algoritmo. Aspectos como validação, implementações físicas e generalizações deste algoritmo também são apresentados;

4. [Guedes et al. 2010a]: Artigo eletrônico contendo exemplos detalhados de ataques quânticos aos geradores de Blum-Blum-Shub [Blum et al. 1986b] e Kaliski [1988]. Este artigo foi idealizado para exemplificar os ataques descritos no artigo publicado no WECIQ;

Os resultados deste trabalho também contribuem para a missão do IQuanta – Instituto de Estudos em Computação e Informação Quânticas, no qual o trabalho foi desenvolvido. Esta dissertação colabora para a consolidação deste instituto como centro de pesquisa na área ao apresentar mais um resultado positivo acerca da utilização da Computação Quântica.

6.2 Trabalhos Futuros

Nesta seção são apresentadas algumas sugestões de trabalhos futuros. Tais sugestões de trabalhos visam aumentar a eficiência da solução proposta e investigar caminhos em aberto

a partir do trabalho realizado.

Em relação ao ataque, uma melhoria significativa seria aumentar o ganho sob a Computação Clássica no tocante a recuperação do representante do estado interno do gerador Blum-Micali. Atualmente este ganho é quadrático e, idealmente, deveria ser superpolinomial. Esta dificuldade computacional, mesmo sob a perspectiva da Computação Quântica, é decorrente do uso do logaritmo discreto pelo gerador Blum-Micali. Embora haja um algoritmo quântico capaz de resolver este problema em tempo polinomial, descrito na Seção 3.2, foi visto que este algoritmo não pode ser usado diretamente, como abordado na Seção 4.5.2. Assim, para tentar aumentar a eficiência do ataque sugere-se a investigação de dois caminhos possíveis:

1. **Desenvolvimento de um algoritmo quântico para o logaritmo discreto independente de oráculo que demanda, em nível clássico, o conhecimento de um dos elementos do estado interno do gerador**. Se tal procedimento existir, o adversário poderia anexá-lo ao final da etapa de identificação do representante e, com isto, seria capaz de recuperar, em tempo polinomial, o estado interno do gerador. Este é um caminho bastante difícil, especialmente considerando que os últimos avanços na elaboração de novos algoritmos quânticos têm sido escassos [Shor 2003];

2. **Investigação da existência de padrões nos grafos funcionais do gerador de Blum-Micali**. A existência de padrões, verificáveis em tempo polinomial, poderia permitir que o adversário modificasse a estratégia de ataque, habilitando a construção de um algoritmo análogo mais eficiente. A investigação de tais padrões possui relação direta com o problema do logaritmo discreto e demanda esforços conjuntos da Computação e da Matemática. A dissertação de Gregg [2003] compila um conjunto de heurísticas para o problema do logaritmo discreto e o artigo de Cloutier e Holden [2010] explora mapeamentos do logaritmo discreto. Estes trabalhos podem ser consultados como ponto de partida na elaboração de um trabalho nesta linha.

Uma vez que os trabalhos desta dissertação tiveram como foco o gerador de Blum-Micali, por motivos previamente expostos, a determinação do número de bits para atacar os demais geradores da Construção de Blum-Micali não foi explorada. A construção de provas formais e estudos experimentais poderia auxiliar nesta investigação e na determinação dos recursos

adequados para a melhor elaboração de ataques a estes geradores.

Uma dificuldade encontrada na realização do trabalho foi a decomposição das portas do tipo $g^{()}$ e δ_{b_i} em portas quânticas básicas, advindas de um conjunto universal de portas quânticas. Esta dificuldade resulta de dois fatores: (i) a ausência de uma caso geral para tais portas, ou seja, a definição destas é feita em função do gerador sob ataque; e (ii) a ausência de um procedimento computacional automatizado para realização desta tarefa. Este último ponto, em particular, é bastante crítico, pois, por exemplo, para uma matriz de ordem 8, como usada na definição da porta $g^{()}$ da Eq. (4.22), são necessárias até 28 matrizes unitárias de dois níveis para sua decomposição. Levando em conta a quantidade de matrizes, há ainda a necessidade de efetuar multiplicações e codificações de Gray, que inviabilizam a realização manual desta tarefa.

Seguindo esta linha, duas sugestões de trabalhos futuros são apresentadas. A primeira delas é investigar a quantidade de portas básicas para implementação das operações $g^{()}$ e δ_{b_i}. Esta investigação aumenta a compreensão do ataque proposto, principalmente em relação a sua viabilidade e recursos necessários para sua implementação. A segunda sugestão diz respeito a construção de aplicativos que automatizem a decodificação de portas quânticas em portas quânticas mais básicas. Tal procedimento eliminaria a realização manual desta operação, que é uma tarefa exaustiva. Além disso, este aplicativo poderia auxiliar em uma análise mais complexa dos algoritmos quânticos atualmente propostos e até ser incorporada em simuladores de circuitos quânticos, favorecendo, inclusive, otimizações na realização de determinadas simulações.

Como última sugestão, a abrangência do ataque proposto a outros geradores, em particular aos demais CSPRNGs, é um aspecto desejável, especialmente levando em consideração os ganhos já verificados nesta dissertação. Um trabalho posterior poderia verificar a possibilidade de adaptar o ataque proposto aos geradores criptograficamente seguros baseados em funções *one-way*, cuja definição encontra-se no trabalho de Håstad et al. [1999]. Tais geradores baseiam-se em problemas para os quais não se conhece solução de tempo polinomial na Computação Clássica. A verificação de ganhos com a Computação Quântica neste cenário seria mais um resultado positivo e motivador para a elaboração de novos ataques a sistemas criptográficos utilizando este paradigma computacional.

Referências Bibliográficas

Ambainis, A. (2004). Quantum search algorithms. *SIGACT News*, 35:22–35.

Bacon, D. and van Dam, W. (2010). Recent Progress in Quantum Algorithms. *Commun. ACM*, 53(2):84–93.

Benioff, P. (1980). The Computer as a Physical System: A Microscopic Quantum Mechanical Hamiltonian Model of Computers as represented by Turing Machines. *Journal of Statistical Physics*, 22:563–591.

Bennett, C. (1973). Logical reversibility of computation. *IBM Journal of Research and Development*, 17:525–532.

Bennett, C., Bernstein, E., Brassard, G., and Vazirani, U. (1997). The Strengths and Weaknesses of Quantum Computation. *SIAM Journal on Computing*, 26(5):1510–1523.

Blum, L., Blum, M., and Shub, M. (1986a). A Simple Unpredictable Pseudo-Random Number Generator. *SIAM Journal on Computing*, 15:364–383.

Blum, L., Blum, M., and Shub., M. (1986b). A Simple Unpredictable pseudo-random number generator. *SIAM Journal on Computing*, pages 364–383.

Blum, M. and Micali, S. (1984). How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*, 13:850–864.

Boyar, J. (1989). Inferring Sequences Produced by Pseudo-Random Number Generators. *Journal of the Association for Computing Machinery*, 36:139–141.

Boyer, M., Brassard, G., Høyer, P., and Tapp, A. (1998). Tight Bounds on Quantum Searching. *Fortsch. Phys. – Prog. Phys.*, 46(4-5):493–505.

Chen, G., Fulling, S. A., Lee, H., and Scully, M. O. (2001). Grover's algorithm for multiobject search in quantum computing. *Lecture Notes in Physics*, 561/2001:165–175.

Chuang, I. (2007). Quantum gates and circuits. Notas de Aula. Disponível em www.cs.berkeley.edu/˜vazirani/f97qcom/lec2.ps.

Cloutier, D. R. and Holden, J. (2010). Mapping the discrete logarithm. *Involve*, 3:197–213.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to Algorithms*. MIT Press.

Davis, D., Ihaka, R., and Fenstermacher, P. (1994). Cryptographic randomness from air turbulence in disk drives. In *CRYPTO '94: Proceedings of the 14th Annual International Cryptology Conference on Advances in Cryptology*, pages 114–120, London, UK. Springer-Verlag.

Delfs, H. and Knebl, H. (2007). *Introduction to Cryptography – Principles and Applications*. Springer.

Deutsch, D. (1985). Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer. *Proceedings of the Royal Society of London A*, 400:97–117.

Deutsch, D. (1989). Quantum computational networks. *Proc. R. Soc. London A*, 425:73–90.

Dirac, P. (1982). *The principles of Quantum Mechanics*. Oxford UK, 4th edition. ISBN 0198520115.

Eastlake, D., Schiller, J., and Crocker, S. (2005). Randomness Requirements for Security. Technical report, Network Working Group – Request for Comments 4086.

Feynman, R. (1982). Simulating Physics with Computers. *International Journal of Theoretical Physics*, 21:467–488.

Gennaro, R. (2005). An improved pseudo-random generator based on the discrete logarithm problem. *Journal of Cryptology*, 18(2):91–110.

Gentle, J. E. (2003). *Random Number Generation and Monte Carlo Methods*. Springer.

Goldreich, O. (2005). *Foundations of Cryptography – A Primer*. now Publishers Inc.

Gregg, J. A. (2003). On factoring integers and evaluating discrete logarithms. Master's thesis, Harvard College.

Grover, L. K. (1997). Quantum Mechanics Helps in Searching for a Needle in a Haystack. *Physical Review Letters*, 79:325–328.

Guedes, E. B. (2009). Sieve – uma aplicação para analisar a qualidade de geradores de números aleatórios e pseudo-aleatórios. Relatório de estágio integrado, Unversidade Federal de Campina Grande.

Guedes, E. B., de Assis, F. M., and Lula Jr., B. (2010a). Examples of the Generalized Quantum Permanent Compromise Attack to the Blum-Micali Construction. http://arxiv.org/abs/1012.1776.

Guedes, E. B., de Assis, F. M., and Lula Jr., B. (2010b). A Generalized Quantum Permanent Compromise Attack to the Blum-Micali Construction. In *III Workshop-Escola de Computação e Informação Quânticas*.

Guedes, E. B., de Assis, F. M., and Lula Jr., B. (2010c). Quantum Permanent Compromise Attack to Blum-micali Pseudorandom Generator. In *International Telecommunications Symposium*.

Guedes, E. B., de Assis, F. M., and Lula Jr., B. (2011). Busca Quântica. Technical report, Universidade Federal de Campina Grande.

Guedes, E. B. and Lula Jr., B. (2010). *Autômatos Finitos – com uma introdução aos Autômatos Finitos Quânticos*. Editora da Universidade Federal de Campina Grande.

Hirvensalo, M. (2001). *Quantum Computing*. Springer.

Hoffstein, J., Pipher, J., and Silverman, J. H. (2008). *An Introduction to Mathematical Cryptography*. Springer Publishing Company, Incorporated.

Håstad, J., Impagliazzoy, R., Levinz, L. A., and Luby, M. (1999). A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396.

Håstad, J., Schrift, A., and Shamir, A. (1993). The discrete logarithm modulo a composite hides $o(n)$ bits. *Journal of Computer and System Sciences*, 47:376–404.

Imre, S. and Balazs, F. (2005). *Quantum Computing and Communications - An Engineering Approach*. John Wiley & Sons.

Isidro, C. R. G. (2008). Uma Abordagem Quântica para o Uso de Expressões Regulares. Master's thesis, Universidade Federal de Campina Grande.

Jain, R. (1991). *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modeling*. John Wiley.

Jozsa, R. (2001). Quantum factoring, discrete logarithms, and the hidden subgroup problem. *Computing in Science and Engg.*, 3(2):34–43.

Kaliski, B. S. (1988). *Elliptic Curves and Cryptography: A Pseudorandom Bit Generator and Other Tools*. PhD thesis, MIT, Cambridge, MA, USA.

Kang, J.-S. (2005). Security Frameworks for Pseudorandom Number Generators. *Trends in Mathematics*, 8(1):1–11.

Kaye, P., Laflamme, R., and Mosca, M. (2007). *An Introduction to Quantum Computing*. Oxford University.

Kelsey, J., Schneider, B., Wagner, D., and Hall, C. (1998). Cryptanalytic attacks on pseudorandom number generators. *Lecture Notes in Computer Science*, 1372/1998:168–188.

Krawczyk, H. (1992). How to Predict Congruential Generators. *Journal of Algorithms*, 13:527–545.

L'Ecuyer, P. (1990). Random numbers for simulation. *Communications of the ACM*, 33(10):85–97.

Lehmer, D. H. (1951). Mathematical methods in large-scale computing units. *Proceedings of the Second Symposium on Large Scale Digital Computing Machinery*, 1:141–146.

Lima, A. F. and Lula Jr., B. (2007). *Computação Quântica: noções básicas utilizando a linguagem de circuitos quânticos*. Editora da Universidade Federal de Campina Grande.

Long, D. and Wigderson, A. (1988). The discrete log hides $o(\log n)$ bits. *SIAM Journal on Computing*, 17:363–372.

Marquezino, F. (2010). *Análise, Simulações e Aplicações Algorítmicas de Caminhadas Quânticas*. PhD thesis, Laboratório Nacional de Computação Científica.

Marsaglia, G. (1968). Random numbers fall mainly in the planes. *Proceedings of the National Academy of Sciences*, 61:25–28.

Marsaglia, G. (1995). The Marsaglia Random Number CDROM, including the DIEHARD Battery of Tests of Randomness. Disponível em http://stat.fsu.edu/. Último acesso em Julho de 2009.

Mermin, N. D. (2007). *Quantum Computer Science – An Introduction*. Cambridge University Press.

Nielsen, M. A. and Chuang, I. L. (2005). *Computação Quântica e Informação Quântica*. Bookman.

Oliveira, N. A. (2007). A utilização do algoritmo quântico de busca em problemas da teoria da informação. Master's thesis, Universidade Federal de Campina Grande.

Paar, C. and Pelzl, J. (2010). *Understanding Cryptography*. Springer.

Patel, S. and Sundaram, G. (1998). An efficient discrete log pseudo random generator. In *CRYPTO'98*, volume 1462 of *LNCS*, pages 304–317.

Peralta, R. (1986). Simultaneous security of bits in the discrete log. In Springer, editor, *EUROCRYPT 1985*, volume 219 of *LNCS*, pages 62–72.

Portugal, R., Lavor, C. C., Carvalho, L. M., and Maculan, N. (2004). *Uma Introdução à Computação Quântica*. Sociedade Brasileira de Matemática Aplicada e Computacional.

Proos, J. and Zalka, C. (2004). Shor's discrete logarithm quantum algorithm for elliptic curves. *Quantum Information & Computation*, 3(4):317–344.

Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Leveson, M., Vangel, M., Banks, D., Heckert, A., Dray, J., and Vo, S. (2008). A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Technical report, National Institute of Standards and Technology – Technology Administration – U. S. Department of Commerce.

Shor, P. (1997). Polynomial-time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26:1484–1509.

Shor, P. W. (2003). Why haven't more quantum algorithms been found? *Journal of the ACM*, 50:87–90.

Sidorenko, A. (2008). *Design and Analysis of Provably secure Pseudorandom Generators*. VDM Verlag.

Sidorenko, A. and Schoenmakers, B. (2005). State recovery attacks on pseudorandom generators. In *Western European Workshop on Research in Cryptology*, pages 53–63.

Solovay, R. and Strassen, V. (1977). A Fast Monte-Carlo Test for Primality. *SIAM J. Comput.*, 6 (1):84–85.

Toscani, L. V. and Veloso, P. A. S. (2002). *Complexidade de Algoritmos*. Editora Artmed.

Turing, A. M. (1936). On Computable Numbers, with an Application to the Entscheidungsproblem. *Proc. London Math. Soc*, 2 (42):230–265.

van Tilborg, H. C. (2005). *Encyclopedia Of Cryptography and Security*. Springer.

Williams, C. P. (2011). *Explorations in Quantum Computing*. Springer, 2nd edition.

Williams, C. P. and Clearwater, S. H. (1998). *Explorations in Quantum Computing*. The Eletronic Library of Science.

Woodhull, A. S. and Tanenbaum, A. S. (2008). *Sistemas Operacionais: Projeto e Implementação*. Artmed.

Yao, A. C. (1982). Theory and applications of trapdoor functions. In *IEEE Symposium on Foundations of Computer Science*, pages 80–91.

Yao, A. C. (1993). Quantum circuit complexity. *Annual IEEE Symposium on Foundations of Computer Science*, -:352–361.

Zalka, C. (1998). Simulating quantum systems on a quantum computer. *Proc. R. Soc. London A*, 454(1969):313–322.

Zalka, C. (1999). Grovers quantum searching algorithm is optimal. quant-ph/9711070.

Apêndice A

Conceitos Básicos da Computação Quântica

A *Computação Quântica* é um paradigma computacional que leva em consideração a Física dos dispositivos ao realizar a computação. Foi proposta por David Deutsch [1985], com a descoberta de uma máquina de Turing quântica universal e, desde então, novos algoritmos vêm sendo propostos de acordo com este paradigma, alguns dos quais com ganho em relação às suas contrapartidas da Computação Clássica, paradigma computacional amplamente utilizado nos dias atuais.

A Mecânica Quântica é parte componente da Teoria Quântica e visa dar suporte à uma descrição da natureza quando se leva em consideração a Física das partículas subatômicas, ou Física Quântica. Pode também ser compreendida como um arcabouço matemático para descrever sistemas quânticos isolados, cujo comportamento não pode ser capturado pela Física Clássica [Williams and Clearwater 1998]. Portanto, para definir algoritmos e construir dispositivos para a Computação Quântica é preciso respeitar os postulados da Mecânica Quântica. Tais postulados especificam como se dá a representação, processamento e medição da informação na Computação Quântica.

Neste capítulo é apresentada uma síntese dos conceitos básicos da Computação Quântica. Tais conceitos encontram-se denotados com a notação de Dirac [1982], uma forma concisa de representação dos conceitos da Mecânica Quântica, que acarreta em uma simplificação dos cálculos a serem realizados. Antes da apresentação de tais conceitos, é mostrado um breve histórico do surgimento deste paradigma computacional.

Os conteúdos apresentados neste capítulo foram obtidos e organizados a partir de diversos trabalhos: Nielsen e Chuang [2005], Kaye et al. [2007], Guedes e Lula Jr. [2010], Williams e Clearwater [1998], a dissertação de Isidro [2008] e a tese de Marquezino [2010]. Sugere-se que o leitor consulte as obras mencionadas caso queira expandir os seus conhecimentos sobre os assuntos abordados.

A.1 Breve Histórico

A primeira noção formal de computação foi capturada pela *máquina de Turing* (MT), um modelo computacional que define como os algoritmos da Computação Clássica devem ser construídos e que ajuda a entender quais os limites desta forma de computação. Segundo Turing, este modelo seria capaz de capturar o significado de realizar uma tarefa por meio de algoritmos, i.e., se um algoritmo pode ser realizado em qualquer tipo de sistema físico, então existe um algoritmo equivalente para a MT. Esta afirmação é a denominada *tese de Church-Turing* [Turing 1936].

Embora fosse capaz de descrever algoritmos para diversos problemas, a MT tinha algumas limitações, especialmente em determinados problemas de ordem prática. Por exemplo, considerando a busca de um elemento em um vetor desordenado de comprimento n, a MT necessitaria percorrer, no pior caso, todo o vetor de elementos. Mas, se escolhas aleatórias fossem feitas, na média, o custo para encontrar o elemento desejado cairia para $n/2$ – embora isto não representasse garantias de não percorrer n elementos.

Diante da possibilidade de "acelerar" a resolução de certos problemas, Solovay e Strassen verificaram que a combinação de uma MT com elementos probabilísticos, tais como um gerador de números aleatórios, seria capaz de resolver eficientemente[1] problemas que não têm solução eficiente no modelo original de Turing [Solovay and Strassen 1977]. Este novo modelo proposto por Solovay e Strassen ficou conhecido como *máquina de Turing probabilística* e levou à extensão da tese de Church-Turing para sua versão forte – que qualquer processo algorítmico poderia ser simulado eficientemente por uma MT probabilística.

Na tentativa de aproveitar vantagens de determinados fenômenos físicos, tais como a

[1] Solução eficiente em um modelo computacional diz respeito a existência de um algoritmo de tempo polinomial deste modelo de computação para resolver o dado problema [Toscani and Veloso 2002].

superposição[2] de estados, na aceleração da resolução de determinados problemas, Benioff considerou um modelo de uma MT com uma fita quântica [Benioff 1980]. Embora este fosse o primeiro trabalho a utilizar componentes quânticos, não seria possível afirmar que este modelo era, de fato, quântico – por utilizar estados clássicos, ao ler a fita quântica, a máquina de Benioff fazia com que os estados da fita também se tornassem clássicos. Em virtude disto, a máquina de Benioff era equivalente a uma MT clássica reversível.

Ao tentar simular sistemas quânticos arbitrários, Richard Feynman, em 1982, provou que nenhuma MT conhecida poderia simular este tipo de sistema forma eficiente, pois tais simulações poderiam culminar em uma aumento exponencial no número de estados, tornando intratável a realização da computação [Feynman 1982]. Este trabalho incentivou a busca por um modelo computacional que superasse esta limitação.

Em 1985, David Deutsch propôs um modelo computacional baseado inteiramente na Física Quântica – a Máquina de Turing Quântica [Deutsch 1985]. O modelo proposto por Deutsch vêm desencadeando desafios à tese forte de Church-Turing, pois algoritmos quânticos eficientes foram propostos para problemas que não se conhece solução eficiente na MT probabilística.

Em paralelo à evolução teórica dos modelos computacionais, nos últimos 40 anos houve uma drástica miniaturização dos componentes computacionais, de acordo com a qual também é possível prever que potencialmente a escala atômica será atingida. A Figura A.1 ilustra esta evolução ao longo do tempo e uma perspectiva de miniaturização até a escala mencionada. Deste ponto de vista, é impossível ignorar os efeitos da Física Quântica, demandando, portanto, o uso da Mecânica Quântica na concepção de algoritmos e dispositivos computacionais.

Portanto, define-se a *Computação Quântica* como o paradigma computacional que leva em conta a física dos dispositivos, ou seja, a Física Quântica, na realização da computação. Diversos algoritmos vêm sendo propostos de acordo com este paradigma. A busca em uma base de dados desordenada [Grover 1997] e a fatoração [Shor 1997] foram os algoritmos de maior destaque propostos. Em particular, o algoritmo de Shor poderá se consolidar como uma ameaça à criptografia de chave assimétrica quando computadores quânticos se tornarem escaláveis.

[2] O conceito de superposição é apresentado na Seção A.2

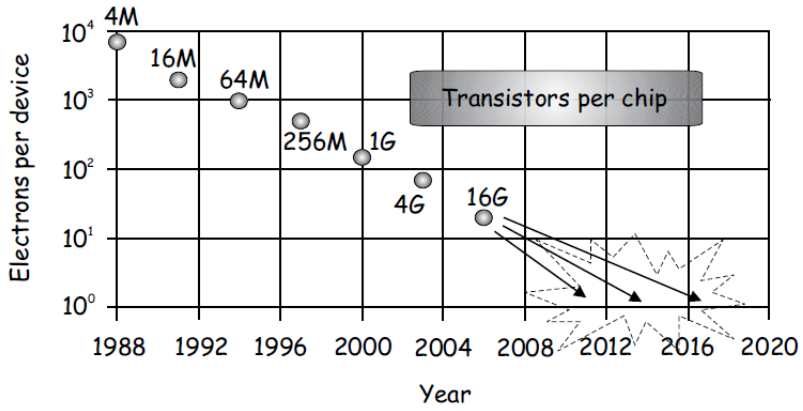

Figura A.1: Perspectiva de miniaturização dos componentes computacionais. O eixo x ilustra o ano e o eixo y a quantidade de elétrons por dispositivo [Imre and Balazs 2005]

Além destes algoritmos, diversos elementos computacionais conhecidos têm sido "atualizados" para o paradigma quântico com ganhos em relação às suas versões clássicas. São exemplos destes elementos: autômatos finitos, caminhantes aleatórios, algoritmos genéticos, redes neurais, gramáticas, dentre outros. Uma síntese dos recentes progressos nesta área pode ser encontrada no artigo de Bacon e van Dam [2010].

A.2 Representação da Informação

Na Computação Clássica, a unidade básica de informação é o bit (*binary digit*), que assume valor 0 ou 1 (falso ou verdadeiro, respectivamente). A representação de uma informação é feita por meio da sua codificação em uma seqüência finita de bits.

Na Computação Quântica, a unidade básica de representação da informação é um sistema quântico de dois estados: o *qubit* (*quantum bit*) . Um qubit $|\psi\rangle$ é representado por meio de um vetor bidimensional em um espaço de Hilbert, de acordo com a seguinte expressão geral:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{A.1}$$

em que α e β são números complexos ($\alpha, \beta \in \mathbb{C}$) e que devem satisfazer à restrição de unitariedade:

$$|\alpha|^2 + |\beta|^2 = 1 \tag{A.2}$$

Nos casos em que os valores de α e β na Eq. (A.1) são diferentes de zero simultaneamente ($\alpha, \beta \neq 0$), diz-se que o qubit está em uma *superposição* destes estados. Quando um qubit está em superposição, não é possível afirmar se este qubit está em $|0\rangle$ ou $|1\rangle$ – estados da base computacional. Para os casos particulares em que $|\alpha| = |\beta|$, diz-se que o qubit está em uma *superposição igualmente distribuída* de estados. Nos exemplos em (A.3) e em (A.4), $|\varphi\rangle$ e $|\phi\rangle$ representam qubits em superposição, mas apenas $|\phi\rangle$ encontra-se em uma superposição igualmente distribuída de estados:

$$|\varphi\rangle = \frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}|1\rangle \tag{A.3}$$

$$|\phi\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle \tag{A.4}$$

De acordo com a notação de Dirac, os estados quânticos (qubits) são representados por *kets* ($|\cdot\rangle$) ou *bras* ($\langle\cdot|$). Por exemplo, os kets $|0\rangle$ e $|1\rangle$, estados da base computacional correspondentes aos bits 0 e 1, são representados da seguinte forma:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{A.5}$$

e o estado geral de um qubit $|\psi\rangle$ tem a seguinte notação:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{A.6}$$

$$= \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{A.7}$$

$$= \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \tag{A.8}$$

Diz-se que α e β são as *amplitudes* associadas aos $|0\rangle$ e $|1\rangle$, respectivamente.

O conjugado transposto de um ket $|\psi\rangle$, denomina-se *bra* e é denotado da seguinte forma, $\langle\psi| = |\psi\rangle^\dagger$ em que $\dagger$ denota duas operações: a operação de conjugado de números complexos e a transposição de uma matriz:

$$\begin{aligned}
\langle\psi| &= |\psi\rangle^{\dagger} && \text{(A.9)}\\
&= (|\psi\rangle^{*})^{T} && \text{(A.10)}\\
&= \begin{bmatrix} \alpha^{*} \\ \beta^{*} \end{bmatrix}^{T} && \text{(A.11)}\\
&= \begin{bmatrix} \alpha^{*} & \beta^{*} \end{bmatrix} && \text{(A.12)}
\end{aligned}$$

Por exemplo, o conjugado transposto do ket $|\varphi\rangle$ mostrado em (A.3) é denotado por:

$$\begin{aligned}
\langle\varphi| &= |\varphi\rangle^{\dagger} && \text{(A.13)}\\
&= \begin{bmatrix} \frac{1}{\sqrt{3}} \\ \sqrt{\frac{2}{3}} \end{bmatrix}^{\dagger} && \text{(A.14)}\\
&= \begin{bmatrix} \frac{1}{\sqrt{3}}^{*} & \sqrt{\frac{2}{3}}^{*} \end{bmatrix} && \text{(A.15)}\\
&= \begin{bmatrix} \frac{1}{\sqrt{3}} & \sqrt{\frac{2}{3}} \end{bmatrix} && \text{(A.16)}
\end{aligned}$$

Uma distinção importante em relação à Computação Clássica é que enquanto um bit pode assumir apenas dois valores distintos, um qubit pode assumir infinitos estados, desde que a restrição de unitariedade (descrita na Equação A.2) seja respeitada. Isto significa que a unidade básica de informação na Computação Quântica é ilimitada, enquanto que a unidade básica da Computação Clássica é limitada aos valores "verdadeiro" e "falso".

A.2.1 Interpretação Geométrica

Uma interpretação possível para um qubit é a interpretação geométrica, na qual um qubit é visto como um vetor contido em uma esfera ($\mathbb{R}^3$). De acordo com esta notação, a expressão geral de um qubit pode ser denotada como segue:

$$|\psi\rangle = e^{i\gamma}\left(\cos(\theta/2)\,|0\rangle + e^{i\varphi}\sin(\theta/2)\,|1\rangle\right) \qquad \text{(A.17)}$$

em que $\gamma, \theta, \varphi \in \mathbb{R}$. O termo $e^{i\gamma}$ é denominado *fator de fase global* e não possui efeito físico observável (uma vez que $|e^{i\gamma}| = 1$) sendo, portanto, preferível a seguinte notação:

$$|\psi\rangle = \cos(\theta/2)\,|0\rangle + e^{i\varphi}\sin(\theta/2)\,|1\rangle \tag{A.18}$$

em que $\theta \in [0, \varphi]$ e $\varphi \in [0, 2\pi]$ denotam um ponto em uma esfera tridimensional de raio 1, conhecida como *esfera de Bloch*. A ilustração desta representação pode ser visualizada na Figura A.2. Esta representação é importante porque auxilia, por exemplo, na interpretação do resultado de operações, algoritmos, etc.

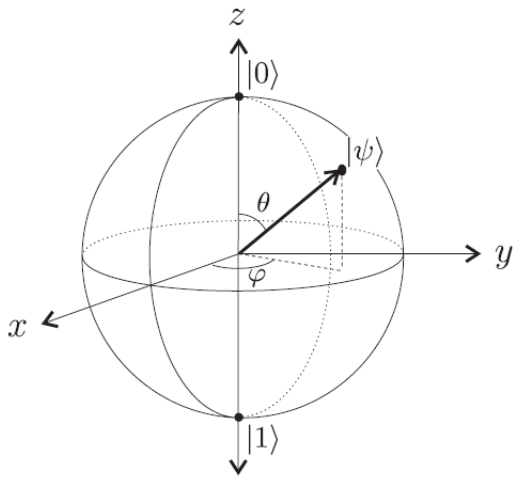

Figura A.2: Estado de um qubit na esfera de Bloch.

Por exemplo, o qubit $|\phi\rangle$ da Eq. (A.4), pode ser denotado, de acordo com a interpretação geométrica, como:

$$|\phi\rangle = \cos\left(\frac{\pi}{4}\right)|0\rangle - \sin\left(\frac{\pi}{4}\right)|1\rangle \tag{A.19}$$

A.2.2 Produto Tensorial de Qubits

No caso da Computação Clássica, um bit é capaz de representar dois valores e um conjunto de n bits, ou registrador de n bits, pode representar 2^n valores diferentes, um por vez. Na Computação Quântica, um registrador de n qubits também pode armazenar 2^n valores, porém todos ao mesmo tempo, graças à superposição. A notação para representar estes registradores quânticos – denominados multi-qubit – faz uso do produto tensorial, denotado por $\otimes$. O produto tensorial de dois qubits $|a\rangle$ e $|b\rangle$, denotado por $|a\rangle \otimes |b\rangle = |ab\rangle = |a\rangle\,|b\rangle$, é mostrado a seguir:

$$|a\rangle \otimes |b\rangle = \begin{bmatrix} a_1 \\ a_2 \\ \cdots \\ a_n \end{bmatrix} \otimes \begin{bmatrix} b_1 \\ b_2 \\ \cdots \\ b_n \end{bmatrix} \tag{A.20}$$

$$= \begin{bmatrix} a_1 \cdot |b\rangle \\ a_2 \cdot |b\rangle \\ \cdots \\ a_n \cdot |b\rangle \end{bmatrix} \tag{A.21}$$

Para exemplificar, seja $|\psi\rangle$ o estado de um sistema de dois qubits:

$$|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \tag{A.22}$$

$$= (\alpha_1 |0\rangle + \beta_1 |1\rangle) \otimes (\alpha_2 |0\rangle + \beta_2 |1\rangle) \tag{A.23}$$

$$= \alpha_1\alpha_2 |00\rangle + \alpha_1\beta_2 |01\rangle + \beta_1\alpha_2 |10\rangle + \beta_2\beta_2 |11\rangle \tag{A.24}$$

$$= \alpha_{00} |00\rangle + \alpha_{01} |01\rangle + \alpha_{10} |10\rangle + \alpha_{11} |11\rangle \tag{A.25}$$

$$= \alpha'_0 |0\rangle + \alpha'_1 |1\rangle + \alpha'_2 |2\rangle + \alpha'_3 |3\rangle \tag{A.26}$$

$$= \sum_{i=0}^{2^2-1} \alpha'_i |i\rangle \tag{A.27}$$

É interessante notar que, do passo (A.25) para o passo (A.26), a notação binária foi substituída pela notação decimal – este é um recurso freqüentemente adotado para promover uma simplificação. O estado $|\psi\rangle$ de dois qubits, contém os estados 0, 1, 2 e 3 ao mesmo tempo, cada um deles com sua amplitude α'_i associada. Neste caso, se todos os $\alpha'_i \neq 0$, então o $|\psi\rangle$ está em superposição. Caso esta mesma informação fosse ser representada em um Computador Clássico, seriam necessários quatro registradores, ao invés de um único como é feito na Computação Quântica.

De forma genérica, o estado geral de um sistema de n qubits pode ser denotado como segue:

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle \tag{A.28}$$

com $\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1$ e a base computacional $\{|0\rangle, |1\rangle, \ldots, |2^n - 1\rangle\}$.

A.3 Processamento da Informação

Na Computação Clássica, o processamento da informação é realizado pela aplicação de operações aos bits que armazenam a informação. Na Computação Quântica, o processamento da informação também é realizado por meio de operadores, denotados tipicamente por letras maiúsculas do alfabeto. Um sistema quântico isolado (qubit) originalmente no estado $|\psi_1\rangle$ *evolui* para o estado $|\psi_2\rangle$ por meio da aplicação de um operador U:

$$|\psi_2\rangle = U\,|\psi_1\rangle \tag{A.29}$$

Os operadores da Computação Quântica são unitários, pois preservam a norma dos vetores, e devem possuir a seguinte propriedade:

$$U \cdot U^\dagger = U^\dagger \cdot U = \mathbb{I} \tag{A.30}$$

em que † denota o conjugado transposto e $\mathbb{I}$ denota a matriz identidade. Por serem unitários, os operadores quânticos também são *reversíveis*. Por exemplo, para o estado $|\psi_2\rangle$, denotado em (A.29), existe um operador $U^\dagger$, inverso de U, tal que:

$$U^\dagger\,|\psi_2\rangle = |\psi_1\rangle \tag{A.31}$$

Várias definições de operadores unitários são possíveis na Computação Quântica. Dentre estas definições, destacam-se as matrizes de Pauli:

$$\mathbb{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{A.32}$$

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \qquad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{A.33}$$

O operador X, em particular, merece destaque, pois é o análogo quântico da porta NOT clássica. Quando o operador X é aplicado ao qubit $|1\rangle$, por exemplo, produz o seguinte resultado:

$$X|1\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{A.34}$$

$$= \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{A.35}$$

$$= |0\rangle \tag{A.36}$$

Outro operador importante para a Computação Quântica porque sua aplicação é capaz de produzir superposições igualmente distribuídas é o *operador de Hadamard*, denotado por H cuja representação matricial é dada a seguir:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{A.37}$$

O operador de Hadamard aplicado aos qubits $|0\rangle$ e $|1\rangle$, por exemplo, cria as seguintes superposições:

$$H|0\rangle = \frac{1}{\sqrt{2}} \cdot |0\rangle + \frac{1}{\sqrt{2}} \cdot |1\rangle = |+\rangle \tag{A.38}$$

$$H|1\rangle = \frac{1}{\sqrt{2}} \cdot |0\rangle - \frac{1}{\sqrt{2}} \cdot |1\rangle = |-\rangle \tag{A.39}$$

$$\tag{A.40}$$

Os estados $|+\rangle$ e $|-\rangle$ compõem uma base denominada *base de Hadamard*, tal que $B_H = \{|+\rangle, |-\rangle\}$.

A.3.1 Produto Tensorial de Operadores

Para que os operadores quânticos possam ser aplicados em sistemas multi-qubit, é necessário que a atuação de operadores também seja estendida por meio do produto tensorial. Sejam então A um operador quântico de dimensões $m \times n$ e B outro operador cuja dimensão é $p \times q$. O produto tensorial $A \otimes B$ resulta em:

$$A \otimes B \equiv \begin{bmatrix} A_{11}B & A_{12}B & \dots & A_{1n}B \\ A_{21}B & A_{22}B & \dots & A_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1}B & A_{m2}B & \dots & A_{mn}B \end{bmatrix} \tag{A.41}$$

O operador resultante, $A \otimes B$, possui dimensões $n \cdot q \times m \cdot p$ e pode então ser aplicado a um produto tensorial de vetores de modo já conhecido. Quando deseja-se denotar o produto tensorial de um operador U qualquer por ele mesmo, abrevia-se a notação para $U^{\otimes n}$, em que n é o número de vezes que o produto tensorial é efetuado. Por exemplo, seja um estado $|\psi\rangle = |00\rangle = |0\rangle^{\otimes 2}$ de dois qubits e, aplicando o operador $H \otimes H = H^{\otimes 2}$ a este estado, tem-se:

$$\begin{aligned} H^{\otimes 2}|0\rangle^{\otimes 2} &= H|0\rangle \otimes H|0\rangle && \text{(A.42)} \\ &= \left[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right] \otimes \left[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right] && \text{(A.43)} \\ &= |+\rangle|+\rangle && \text{(A.44)} \\ &= |+\rangle^{\otimes 2} && \text{(A.45)} \end{aligned}$$

O operador de Hadamard, em particular, é bastante útil em diversos algoritmos da Computação Quântica, pois ao criar uma superposição igualmente distribuída de estados, permite que qualquer operador seja aplicado a todos os estados da superposição de forma simultânea, graças ao *paralelismo quântico*. Este paralelismo não é realizado eficientemente pelos computadores clássicos que, para simular uma superposição igualmente distribuída de n qubits, necessitaria de 2^n registradores clássicos nos quais a operação desejada deveria ser repetida em cada um deles de forma seqüencial.

A.3.2 Operadores de Projeção

Quando se efetua o produto externo de um vetor por ele mesmo define-se um tipo especial de operador denominado *operador de projeção*. Os operadores de projeção quando aplicados a um vetor de um espaço vetorial, projetam este vetor em um subespaço vetorial do espaço original. O subespaço vetorial em questão é definido pelos autovalores do vetor que definiu o operador de projeção.

Por exemplo, seja o operador de projeção $|0\rangle\langle 0|$, que possui a seguinte definição:

$$|0\rangle\langle 0| = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \tag{A.46}$$

Quando aplicado ao estado $|+\rangle$, por exemplo, tem-se:

$$|0\rangle\langle 0|+\rangle = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \tag{A.47}$$

$$= \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \end{bmatrix} \tag{A.48}$$

$$= \frac{1}{\sqrt{2}}|0\rangle \tag{A.49}$$

Neste caso, é importante salientar que a relação (A.2) não é respeitada, pois o resultado desta operação é um vetor pertencente a um subespaço do espaço de Hilbert original. Se for considerada a projeção do qubit $|+\rangle$ com o operador $|1\rangle\langle 1|$, advindo do $|1\rangle$ da base computacional, então o resultado é $|1\rangle\langle 1|+\rangle = \frac{1}{\sqrt{2}}$. Se somados, os resultados das duas projeções compõem um qubit do espaço de Hilbert original que, portanto, respeita a relação de unitariedade.

Uma propriedade importante sobre operadores é a *relação de completude*. A relação de completude estabelece que o somatório de todos os projetores obtidos a partir de vetores de uma base de um espaço vetorial é igual à matriz identidade.

Por exemplo, seja a base de Hadamard, B_H. Os operadores de projeção obtidos a partir desta base são:

$$P_+ = |+\rangle\langle +| = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} \tag{A.50}$$

$$P_- = |-\rangle\langle -| = \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \tag{A.51}$$

Para verificar que a relação de completude é obedecida, basta observar que:

$$P_+ + P_- = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} + \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \tag{A.52}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{A.53}$$

$$= \mathbb{I} \tag{A.54}$$

A.4 Medição da Informação

Um sistema quântico isolado possui sua evolução descrita por transformações unitárias. Mas, para acessar o estado de um sistema quântico é necessário realizar uma tarefa denominada *medição*. A medição é uma "interface" entre os níveis quântico e clássico, caracterizando-se como um meio de extrair informações úteis de qubits após determinado processamento.

Na Computação Clássica a extração de informação do estado de bits é conceitualmente simples e, portanto, não costuma ser considerada como uma parte do processo de computação. Porém, na Computação Quântica a medição é uma tarefa não-trivial, pois afeta o sistema quântico isolado, provocando um *colapso* (redução) no espaço de estados deste sistema após a medição. Em virtude disto, a medição é a única operação irreversível na Computação Quântica – uma vez que um qubit é medido, não há meios de fazê-lo voltar ao estado anterior à medição [Mermin 2007].

No escopo deste trabalho, a medição de interesse é a *medição projetiva*, definida como segue: seja o conjunto de projetores $\{M_m\}$ que atuam sobre o espaço de estados de um sistema quântico ou, equivalentemente, o conjunto de todos os projetores geradores pelos vetores uma base. O índice m refere-se aos possíveis resultados da medição. Se o estado de um sistema quântico for $|\psi\rangle$, imediatamente antes da medição, então a probabilidade $p(m)$ do valor m ocorrer como resultado da aplicação dos operadores de medição é dada por:

$$p(m) = \langle\psi| M_m^\dagger M_m |\psi\rangle \tag{A.55}$$

e o estado $|\psi'\rangle$ do sistema após a medição será

$$|\psi'\rangle = \frac{M_m |\psi\rangle}{\sqrt{\langle\psi| M_m^\dagger M_m |\psi\rangle}} \tag{A.56}$$

Para exemplificar a medição, seja um qubit genérico $|\psi\rangle$:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{A.57}$$

Suponha que se deseje efetuar uma medição projetiva neste qubit utilizando projetores da base computacional, ou seja, $\{|0\rangle\langle 0|, |1\rangle\langle 1|\}$. A medição forçará o qubit a colapsar para $|\psi\rangle \to |0\rangle$ ou para $|\psi\rangle \to |1\rangle$, com probabilidades $|\alpha|^2$ e $|\beta|^2$, respectivamente. Ao consultar um display digital com os resultados da medição, por exemplo, serão vistos os valores 0 ou 1, para $|0\rangle$ e $|1\rangle$, respectivamente, mas o estado original (A.57) será perdido e também não será possível conhecer os valores de α e β. Os efeitos da medição com projetores da base computacional em um qubit genérico são descritos na Tabela A.1.

Tabela A.1: Efeitos da medição em um qubit

Estado Inicial	Medição		
	Valor Medido	Estado Final	Probabilidade Associada
$\|\varphi\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$	0	$\|0\rangle$	$\|\alpha\|^2$
	1	$\|1\rangle$	$\|\beta\|^2$

Para exemplificar uma medição, seja o qubit $|\varphi\rangle = \frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}|1\rangle$. A probabilidade deste qubit assumir o estado $|0\rangle$ após a medição é dada por:

$$\begin{aligned}
p(0) &= \langle\varphi|\,|0\rangle\langle 0|^{\dagger}\,|0\rangle\langle 0|\,|\varphi\rangle && \text{(A.58)}\\
&= \langle\varphi|\,|0\rangle\langle 0|\,|0\rangle\langle 0|\,|\varphi\rangle && \text{(A.59)}\\
&= \langle\varphi|\,|0\rangle\langle 0|\,|\varphi\rangle && \text{(A.60)}\\
&= \left(\frac{1}{\sqrt{3}}\langle 0| + \sqrt{\frac{2}{3}}\langle 1|\right)|0\rangle\langle 0|\left(\frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}|1\rangle\right) && \text{(A.61)}\\
&= \left(\frac{1}{\sqrt{3}}\langle 0|\,0\rangle + \sqrt{\frac{2}{3}}\langle 1|\,0\rangle\right)^2 && \text{(A.62)}\\
&= \left(\frac{1}{\sqrt{3}}\right)^2 && \text{(A.63)}\\
&= \frac{1}{3} && \text{(A.64)}
\end{aligned}$$

e a probabilidade deste mesmo qubit assumir o estado $|1\rangle$ é dada por:

$$
\begin{aligned}
p(1) &= \langle\varphi|\,|1\rangle\,\langle 1|^{\dagger}\,|1\rangle\,\langle 1|^{\dagger}\,|\varphi\rangle && \text{(A.65)}\\
&= \langle\varphi|\,|1\rangle\,\langle 1|\,|1\rangle\,\langle 1|\,|\varphi\rangle && \text{(A.66)}\\
&= \langle\varphi|\,|1\rangle\,\langle 1|\,|\varphi\rangle && \text{(A.67)}\\
&= \left(\frac{1}{\sqrt{3}}\,\langle 0| + \sqrt{\frac{2}{3}}\,\langle 1|\right)|1\rangle\,\langle 1|\left(\frac{1}{\sqrt{3}}\,|0\rangle + \sqrt{\frac{2}{3}}\,|1\rangle\right) && \text{(A.68)}\\
&= \left(\frac{1}{\sqrt{3}}\,\langle 0|\,1\rangle + \sqrt{\frac{2}{3}}\,\langle 1|\,1\rangle\right)^{2} && \text{(A.69)}\\
&= \left(\sqrt{\frac{2}{3}}\right)^{2} && \text{(A.70)}\\
&= \frac{2}{3} && \text{(A.71)}\\
&= 1 - p(0) && \text{(A.72)}
\end{aligned}
$$

Embora as probabilidades deste qubit assumir o estado $|0\rangle$ ou $|1\rangle$ sejam conhecidas, não é possível determinar, com 100% de certeza, para qual destes estados o qubit colapsará. Além disso, uma vez que há a medição, o estado $|\varphi\rangle = \frac{1}{\sqrt{3}}\,|0\rangle + \sqrt{\frac{2}{3}}\,|1\rangle$ é perdido.

Notas do Apêndice

Neste capítulo foram apresentados os conceitos básicos da Computação Quântica – paradigma computacional leva em consideração a Física Quântica na realização da computação.

De forma resumida, a representação da informação neste paradigma é feita por qubits, representados por vetores unitários em um espaço de Hilbert. Qubits podem assumir infinitos estados, desde que a condição de unitariedade seja respeitada, e também podem estar em superposição. O processamento da informação é realizado por operadores unitários, que também são reversíveis. Diversos operadores são definidos na Computação Quântica, dentre os quais destacam-se as matrizes de Pauli e o operador de Hadamard. Operadores de projeção, advindos do produto externo de um vetor por ele mesmo, projetam qubits em um subespaço vetorial. Quando um qubit está em superposição é possível aplicar um operador, em único passo, a todos os estados superpostos graças ao paralelismo quântico. A medição

da informação, por sua vez, traz para o domínio clássico os resultados de uma determinada Computação Quântica. Operadores de projeção definem os possíveis resultados e o estado que o sistema computacional assume após a medição, que é a única operação na Computação Quântica que é irreversível.

Apêndice B

Circuitos Quânticos

Circuitos lógicos são utilizados para descrever a atuação de diversos operadores em bits. Esta notação permite expressar a ordem em que as operações são realizadas, quais bits envolvem e os efeitos nos bits de entrada em um computador clássico.

Embora permita uma clara e precisa descrição do que acontecem com os bits ao longo do tempo, a notação dos circuitos lógicos não restringe como tais operações devem ser realizadas fisicamente, ou seja, funciona como um "guia" para a construção de dispositivos computacionais e também como uma forma sintética de descrever algoritmos. Na Figura B.1, há um exemplo de circuito lógico para um somador completo de dois bits, a organização de quatro destes somadores sob a forma de um chip e, finalmente, uma implementação física do mesmo, compondo o chip 7483 amplamente comercializado.

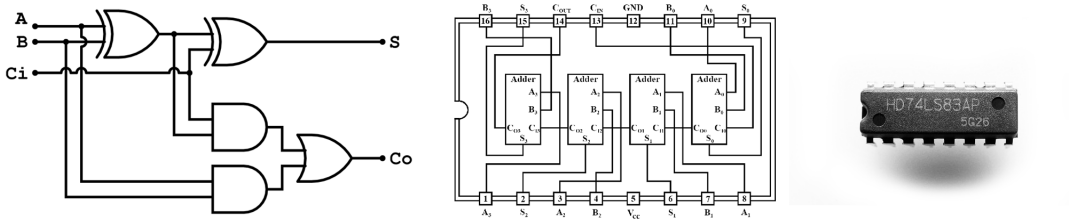

Figura B.1: Exemplo de um circuito lógico para um somador completo de dois bits, sua organização de quatro destes somadores sob a forma de um chip e, finalmente, uma implementação física do mesmo.

Na Computação Quântica, o análogo aos circuitos lógicos da Computação Clássica são os *circuitos quânticos*. Tais circuitos foram propostos por Deutsch [1989] e encapsulam a representação, processamento e medição da informação quântica em uma notação gráfica.

Circuitos quânticos são compostos por portas quânticas, cuja aplicação é sincronizada ao longo do tempo. Assim como os circuitos clássicos, esta notação é adequada para expressar algoritmos e guiar implementações físicas de dispositivos.

Ao longo deste capítulo, a notação de circuitos quânticos será apresentada em detalhes. A Seção B.1 introduz a notação e convenções básicas para a representação de circuitos quânticos, a Seção B.2 apresenta em detalhes a definição, exemplos e tipos de portas quânticas, a Seção B.3 descreve a equivalência entre circuitos quânticos e Máquinas de Turing quânticas e, por fim, na Seção B.4 são mostrados os aspectos da análise de complexidade dos circuitos quânticos.

Os conceitos deste capítulo foram extraídos de diversas obras. Os livros de Lima e Lula Jr. [2007], Kaye et al. [2007] e de Portugal et al. [2004] possuem uma descrição completa dos circuitos quânticos, incluindo a notação e os elementos utilizados. As notas de aula de Chuang [2007] foram utilizadas como referência para apresentação dos conceitos sobre emaranhamento e Teorema da não-clonagem. A equivalência do modelo de circuitos quânticos com a Máquina de Turing quântica é apresentada por Hirvensalo [2001], o qual sintetiza os resultados propostos por Yao [1993]. Por fim, a análise de complexidade dos circuitos quânticos foi descrita tendo como base a Seção 1.3 do livro de Kaye et al. [2007] e também o Apêndice B da dissertação de Isidro [2008]. Estes trabalhos devem ser consultados caso o leitor queira expandir seus conhecimentos.

B.1 Notação e Convenções

O primeiro elemento a se considerar na notação de circuitos quânticos é a *entrada*. Representada na posição mais à esquerda do circuito, denota quais qubits serão processados. A entrada pode ser um único qubit ou um produto tensorial de qubits, os quais podem estar agrupados em um ou mais *registradores* para denotar diferentes partes da entrada ou para distinguir qubits auxiliares dos demais qubits, por exemplo.

As *linhas horizontais* nos circuitos quânticos representam a evolução de um qubit ao longo do tempo e, em virtude disto, não são necessariamente compreendidas como fios. O *sentido* de leitura de um circuito quântico é da esquerda para a direita. Para exemplificar os conceitos ilustrados, a Figura B.1 apresenta um circuito quântico tendo o qubit $|\psi\rangle$ como

entrada. Neste circuito, nenhuma operação é realizada sob a entrada.

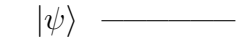

Figura B.2: Circuito quântico simples, cuja entrada é o $|\psi\rangle$.

As *portas quânticas*, denotadas por retângulos, ilustram a realização de operações sobre qubits. Portas quânticas serão apresentadas em detalhes na Seção B.2. Na Figura B.3, a entrada é um produto tensorial $|\psi\rangle \otimes |\varphi\rangle \otimes |\phi\rangle$, e está organizada em dois registradores: o primeiro deles englobando os qubits $|\psi\rangle$ e $|\varphi\rangle$, e o segundo registrador composto apenas do qubit $|\phi\rangle$.

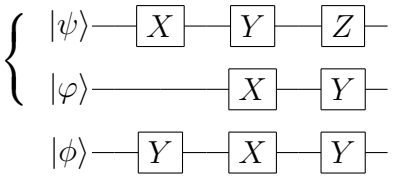

Figura B.3: Circuito quântico com portas X, Y e Z advindas das matrizes de Pauli homônimas. Este circuito possui como entrada os qubits $|\psi\rangle$, $|\varphi\rangle$ e $|\phi\rangle$, organizados em dois registradores.

A *saída* dos circuitos quânticos é a posição mais à direita nestes circuitos. Esta saída pode ser ou não medida. A porta que representa a operação de medição é denotada por um indicador de leitura. A medição que se considera é a medição projetiva na base computacional.

Normalmente quando um qubit é medido, a sua saída quântica é ignorada, sendo apenas de interesse a informação clássica obtida. Em virtude disto, após uma porta de medição são denotados dois fios horizontais, que representam saída de informação clássica [Kaye et al. 2007]. Na Figura B.4, o circuito representado possui portas de medição apenas no primeiro registrador.

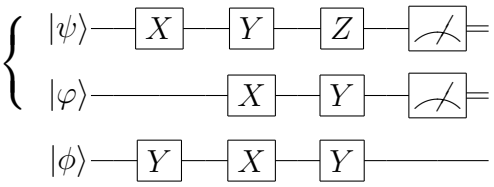

Figura B.4: Circuito quântico da Figura **??**, no qual portas de medição foram inseridas no primeiro registrador.

Para que a compreensão passo-a-passo do que acontece em um circuito quântico seja possível, linhas verticais tracejadas são inseridas após a realização de operações. Estas linhas são abstrações para denotar o estado do sistema quântico em uma unidade de tempo. Para exemplificar a compreensão passo-a-passo, seja o circuito da Figura B.5.

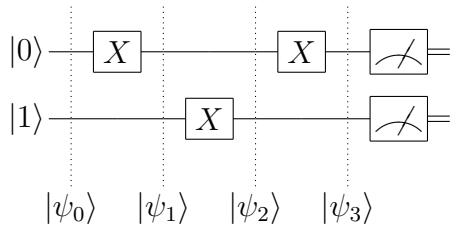

Figura B.5: Circuito quântico com entradas $|0\rangle$ e $|1\rangle$, utilização de portas quânticas X e medições de todos os qubits na saída.

Este circuito é um extremamente simples, pois é composto de uma entrada pequena em estados da base computacional e utiliza apenas uma porta – a porta X. O efeito desta porta é a inversão do qubit, similar à porta NOT clássica, ou seja, se o qubit está no estado $|0\rangle$ assume o estado $|1\rangle$ e vice-versa. Apesar da simplicidade, o acompanhamento dos passos deste circuito é o mesmo utilizado em circuitos mais complexos.

A entrada deste circuito é descrita pelo $|\psi_0\rangle$ que encontra-se no seguinte estado:

$$|\psi_0\rangle = |0\rangle \otimes |1\rangle \tag{B.1}$$

O primeiro passo do circuito é a aplicação da porta X no primeiro qubit. O segundo qubit deve permanecer inalterado, portanto, denota-se a operação $\mathbb{I}$ como a operação que não altera o estado de um qubit. Assim, o estado $|\psi_1\rangle$ é denotado a seguir:

$$\begin{aligned} |\psi_1\rangle &= (X \otimes \mathbb{I})\,|\psi_0\rangle && \text{(B.2)} \\ &= X\,|0\rangle \otimes \mathbb{I}\,|1\rangle && \text{(B.3)} \\ &= |1\rangle \otimes |1\rangle && \text{(B.4)} \end{aligned}$$

É interessante notar que nenhuma alteração foi efetuada no segundo qubit, tal como sugere o circuito. O próximo passo consiste na aplicação do operador X ao estado do segundo qubit.

O estado $|\psi_2\rangle$ segue:

$$|\psi_2\rangle = (\mathbb{I} \otimes X)\,|\psi_1\rangle \tag{B.5}$$

$$= \mathbb{I}\,|1\rangle \otimes X\,|1\rangle \tag{B.6}$$

$$= |1\rangle \otimes |0\rangle \tag{B.7}$$

Por fim, o estado $|\psi_3\rangle$ contempla a aplicação da porta X ao primeiro qubit:

$$|\psi_3\rangle = (X \otimes \mathbb{I})\,|\psi_2\rangle \tag{B.8}$$

$$= X\,|1\rangle \otimes \mathbb{I}\,|0\rangle \tag{B.9}$$

$$= |0\rangle \otimes |0\rangle \tag{B.10}$$

Uma vez que $|\psi_3\rangle$ encontra-se no estado $|0\rangle \otimes |0\rangle$, a medição resultará em 0 e 0 com 100% de certeza. Neste exemplo, foram ilustradas todas as etapas da evolução do estado de dois qubits ao longo das operações realizadas em um circuito quântico.

É importante salientar que em circuitos quânticos não há sentido em haver retroalimentação, que é comum ocorrer em circuitos clássicos, e também não são permitidas bifurcações, nem junções de qubits, pois, reforçando, as linhas horizontais representam o estado de um qubit ao longo do tempo e não fios como em circuitos lógicos clássicos.

B.2 Portas Quânticas

As portas em circuitos lógicos são responsáveis pela implementação de operações em bits. As portas AND, OR e NOT, por exemplo, compõem um conjunto universal de portas que, a partir das quais, é possível implementar qualquer procedimento computacional.

Assim como os circuitos lógicos clássicos, os circuitos quânticos também possuem portas, denominadas *portas quânticas*. Tais portas são responsáveis pela implementação de operações unitárias em qubits de entrada. As operações unitárias são representadas por matrizes e descrevem como um qubit evolui ao longo do tempo. Os conceitos básicos de operações unitárias são apresentados na Seção A.3.

No contexto dos circuitos quânticos, as portas podem ser divididas em categorias, de acordo com a quantidade de qubits que recebem como entrada e também quanto a existência de controles. Tais tipos de portas são apresentados nas seções a seguir.

B.2.1 Portas Quânticas de 1-qubit

No caso da Computação Clássica, a única porta lógica que pode atuar sobre um único bit é a porta NOT. O efeito desta porta é a inversão do valor do bit, ou seja, se o bit é 1 passa a assumir o valor 0 e vice-versa. No caso da Computação Quântica, podem ser definidas infinitas portas que atuam sobre um único qubit. Tais portas são denominadas *portas quânticas de* 1*-qubit* ou *portas quânticas simples*.

As portas quânticas de 1-qubit realizam uma operação unitária U sobre o estado $|\psi\rangle$ de um qubit, fazendo-o evoluir para o estado $U|\psi\rangle$. Toda porta quântica é rotulada com o identificador do operador que implementa, isto permite a fácil identificação da operação em questão. As portas quânticas mais comumente usadas são aquelas advindas das matrizes de Pauli X, Y e Z. A Figura B.6 ilustra a aplicação sucessiva de tais portas seguida de uma medição.

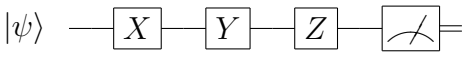

Figura B.6: Circuito quântico com uma única entrada, $|\psi\rangle$, na qual são aplicadas as portas X, Y e Z seguidas de uma medição.

Embora o operador $\mathbb{I}$ seja uma das matrizes de Pauli, este operador não é denotado como porta quântica, uma vez que não produz qualquer efeito sobre o estado do qubit em que é aplicado. Apesar disso, o operador $\mathbb{I}$ não deve ser esquecido em situações em que é preciso aplicar um operador qualquer a apenas parte da entrada, pois a dimensão do espaço de Hilbert deve ser respeitada. As Eqs. (B.2-B.8) exemplificam tais situações.

Existem portas quânticas que não possuem equivalente clássico, a exemplo da porta de Hadamard, denotada por H. Esta porta coloca o qubit de entrada em uma superposição igualmente distribuída de estados. A Figura B.7 ilustra a utilização de tal porta.

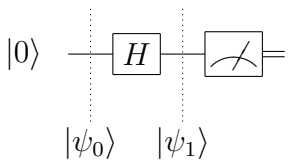

Figura B.7: Circuito quântico demonstrando a utilização da porta de Hadamard.

Para exemplificar a atuação da porta H no circuito da Figura B.7, tem-se que o estado inicial é dado por:

$$|\psi_0\rangle = |0\rangle \tag{B.11}$$

Ao estado inicial é aplicada a porta de Hadamard, produzindo o seguinte efeito:

$$\begin{aligned} |\psi_1\rangle &= H\,|\psi_0\rangle && \text{(B.12)} \\ &= H\,|0\rangle && \text{(B.13)} \\ &= \frac{1}{\sqrt{2}}\,|0\rangle + \frac{1}{\sqrt{2}}\,|1\rangle && \text{(B.14)} \end{aligned}$$

Uma medição neste estado retorna 0 ou 1 de forma equiprovável, pois $\left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2} = 50\%$. Pode-se dizer, portanto, que este circuito quântico, o qual utiliza apenas uma porta, é uma implementação de um gerador de bits aleatórios.

Outra porta quântica de 1-qubit que não possui equivalente clássico é a porta de fase, denotada pela letra S. Esta porta introduz uma fase relativa[1] $e^{i\cdot\frac{\pi}{2}} = i$, em que i é a unidade imaginária. A matriz do operador que esta porta implementa é dado a seguir:

$$S = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\cdot\frac{\pi}{2}} \end{bmatrix} \tag{B.15}$$

Para a construção de algoritmos quânticos complexos, é necessário que a entrada seja composta de vários qubits. Portanto, a noção de portas quânticas precisa ser estendida para levar em consideração a aplicação conjunta de operadores a estas entradas. Os conceitos de portas quânticas multi-qubits são apresentados na seção a seguir.

B.2.2 Portas Quânticas Multi-Qubit

As portas descritas na subseção anterior atuam sobre um qubit apenas. No entanto, para realizar qualquer operação quântica são necessárias também portas que atuem sobre múltiplos qubits.

A primeira forma de estender a noção de portas quânticas de 1-qbuit para portas quânticas multi-qubit é por meio do produto tensorial dos operadores implementados pelas portas de 1-qubit. Por exemplo, para que uma porta U de 1-qubit atue em um estado de n qubits, denotado por $\left|\psi^{(n)}\right\rangle$, basta que efetue-se o produto tensorial deste operador sucessivamente,

[1] Conceitos sobre a interpretação geométrica de qubits são apresentados na Seção A.2.1.

obtendo $U^{\otimes n}$ e, conseqüentemente, a porta correspondente. O produto tensorial de operadores é apresentado na Seção A.3.1. Para ilustrar este tipo de operação, seja a aplicação da porta de Hadamard a três qubits, como mostrado na Figura B.8.

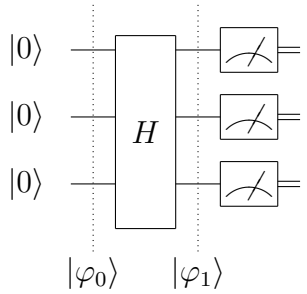

Figura B.8: Circuito quântico demonstrando a utilização da porta de Hadamard.

O estado inicial do circuito é dado por:

$$|\varphi_0\rangle = |0\rangle \otimes |0\rangle \otimes |0\rangle \tag{B.16}$$

$$= |000\rangle \tag{B.17}$$

O estado $|\varphi_1\rangle$ corresponde à aplicação da porta de Hadamard aos três qubits da entrada:

$$|\varphi_1\rangle = H^{\otimes 3} |\varphi_0\rangle \tag{B.18}$$

$$= H|0\rangle \otimes H|0\rangle \otimes H|0\rangle \tag{B.19}$$

$$= \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \tag{B.20}$$

$$= \frac{1}{\sqrt{8}} \left(|000\rangle + |001\rangle + |010\rangle + \ldots + |111\rangle\right) \tag{B.21}$$

$$= \frac{1}{\sqrt{8}} \left(|0\rangle + |1\rangle + |2\rangle + \ldots + |7\rangle\right) \tag{B.22}$$

Uma medição no estado $|\varphi_1\rangle$ resulta em qualquer valor de 0 a 7 de forma equiprovável. Tal efeito faz desta porta um recurso extremamente utilizado por diversos algoritmos quânticos consagrados [Mermin 2007].

Uma das portas de multi-qubits de extrema utilização é a porta $CNOT$ ou NOT-controlada. Esta porta define uma operação sobre dois qubits a e b, denominados *controle* e *alvo*, respectivamente. A porta CNOT atua da seguinte maneira: se o qubit de controle está no estado $|1\rangle$, então o qubit alvo é invertido; se o qubit de controle está no estado $|0\rangle$, então o qubit alvo permanece inalterado. Considerando dois qubits na base computacional, o efeito da porta $CNOT$ é dado como segue:

$$\begin{aligned} |00\rangle &\rightarrow |00\rangle \\ |01\rangle &\rightarrow |01\rangle \\ |10\rangle &\rightarrow |11\rangle \\ |11\rangle &\rightarrow |10\rangle \end{aligned} \tag{B.23}$$

Observando os efeitos desta porta, é possível concluir que a inversão do qubit alvo é análoga à aplicação da porta X condicionada ao qubit de controle. Portanto, o efeito da porta $CNOT$ no qubit alvo pode ser representada por $X^a |b\rangle$, ou ainda por $a \oplus b$, em que $\oplus$ denota a soma módulo 2. A matriz do operador $CNOT$ é mostrada a seguir:

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{B.24}$$

Um circuito contendo esta porta é ilustrado na Figura B.9.

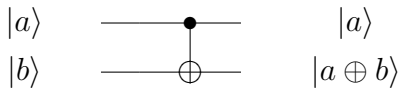

Figura B.9: Circuito quântico contendo uma porta $CNOT$.

A porta quântica $CNOT$ e as portas de 1-qubit compõem um *conjunto de portas quânticas universais*, ou seja, quando combinadas são capazes de implementar quaisquer operação unitária definida sobre qubits.

Uma extensão possível da porta $CNOT$ pode ser definida quando a inversão de um qubit é controlada por dois qubits. Tal extensão define a porta Toffoli quântica, que pode ser vista na Figura B.10.

B.2.3 Portas Quânticas Controladas

É possível definir uma combinação entre portas quânticas e controles. Nesta combinação, se os qubits de controle estão no estado $|1\rangle$, então a porta quântica deve ser aplicada aos qubits

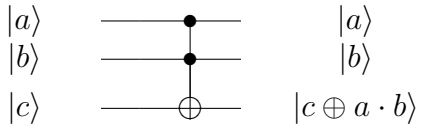

Figura B.10: Circuito quântico contendo uma porta Toffoli quântica.

alvo; caso os controles estejam no estado $|0\rangle$, não há qualquer alteração nos qubits alvo. Para qualquer porta U combinada a um ou mais controles, diz-se que a porta é U-controlada. A Figura B.11 ilustra uma porta U-controlada, em que $|\psi\rangle$ e $|\varphi\rangle$ são, respectivamente, controle e alvo.

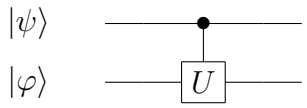

Figura B.11: Circuito quântico contendo uma porta U-controlada.

Também é possível definir que o controle de uma porta quântica seja ativado pelo $|0\rangle$, ou seja: se o controle está no estado $|0\rangle$, então a referida porta quântica é aplicada ao alvo, em caso contrário, o alvo é mantido inalterado. A Figura B.12 ilustra a versão do circuito quântico da Figura B.11 com esta diferenciação na forma de controlar a aplicação da porta quântica.

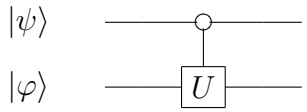

Figura B.12: Circuito quântico denotando uma porta controlada pelo qubit $|0\rangle$.

A utilização de portas quânticas controladas em circuitos que implementam algoritmos é um recurso freqüentemente utilizado. A decisão de aplicar ou não uma determinada porta quântica, a depender do valor do qubit de controle, é similar a um desvio condicional das linguagens de programação.

B.2.4 Emaranhamento

Um dos recursos comuns na Computação Clássica é a cópia de informação. Para construir um circuito quântico capaz de realizar esta tarefa, de copiar um estado quântico genérico, é necessária apenas a utilização de uma porta $CNOT$, como mostrado na Figura B.13.

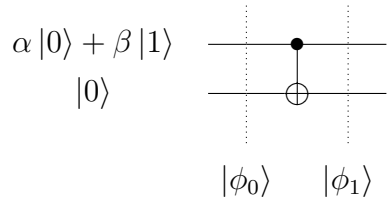

Figura B.13: Circuito quântico para copiar um qubit genérico.

A entrada deste circuito quântico $|\phi_0\rangle$ é dada por:

$$|\phi_0\rangle = |\psi\rangle \otimes |0\rangle \tag{B.25}$$

$$= (\alpha|0\rangle + \beta|1\rangle)|0\rangle \tag{B.26}$$

$$= \alpha|00\rangle + \beta|10\rangle \tag{B.27}$$

A função da porta $CNOT$, como visto, é negar o segundo qubit quando o primeiro encontra-se no estado $|1\rangle$. Assim, a saída é dada por:

$$|\phi_1\rangle = \alpha|00\rangle + \beta|11\rangle \tag{B.28}$$

É interessante observar que ambos os qubits encontram-se no mesmo estado. Se uma medição for realizada em um destes qubits, será observado 0 com probabilidade $|\alpha|^2$ e 1 com probabilidade $|\beta|^2$. Porém, nesta situação, uma vez que um qubit é medido, o estado do outro qubit é alterado, assumindo o mesmo estado do qubit que foi medido. Em tais situações, quando uma medição em um qubit faz um outro qubit assumir o mesmo estado do qubit medido, diz-se que ambos qubits estão *emaranhados*.

Quando qubits estão emaranhados, não é possível ganhar nenhuma informação a respeito de α e β, pois tais dados em ambos os qubits são perdidos com a realização da medição, não podendo ser acessados uma outra vez. Na prática, isto significa que, de fato, os valores de α e β não foram copiados. A impossibilidade de haver cópia de informação quântica é explicada na seção a seguir, pelo Teorema da Não-Clonagem [Chuang 2007].

O emaranhamento, assim como o paralelismo e a superposição, são características exclusivas da Computação Quântica, que promovem ganhos em alguns algoritmos quando comparados aos algoritmos clássicos equivalentes.

B.2.5 Teorema da Não-Clonagem

De acordo com o Teorema da Não-Clonagem, não existe uma porta quântica U capaz de realizar a seguinte operação:

$$U\left|\psi\right\rangle\left|0\right\rangle \rightarrow \left|\psi\right\rangle\left|\psi\right\rangle \tag{B.29}$$

Este teorema respeita o princípio da incerteza de Heisenberg e impõe uma restrição quanto à existência de determinado tipo de portas quânticas [Chuang 2007].

B.3 Equivalência com a Máquina de Turing Quântica

O modelo de circuitos quânticos permite a descrição de qualquer algoritmo quântico, pois este modelo é provadamente equivalente à Máquina de Turing Quântica [Yao 1993; Hirvensalo 2001]. Este resultado possibilita a descrição de algoritmos quânticos apenas por meio de circuitos, uma maneira mais prática de expressar e entender os passos dos algoritmos, particularmente por existir um modelo análogo na Computação Clássica. Esta equivalência também permite expressar algoritmos quânticos, denotados por circuitos, em termos da complexidade de tempo e de espaço, métricas utilizadas pelas Máquinas de Turing Quânticas.

Para que algoritmos quânticos sejam considerados adequados, além de respeitar os postulados da Mecânica Quântica, os circuitos que o implementam devem definir uma *família de circuitos quânticos*. Uma família de circuitos é um conjunto de circuitos $\{C_n | n \in \mathbb{Z}_+\}$, em que há um circuito para cada entrada de tamanho n, ou seja, há um circuito quântico para cada entrada do problema de tamanho diferente. Uma família de circuitos quânticos é dita ser *uniforme* se cada C_n pode ser construído facilmente, isto é, se há uma Máquina de Turing que gera os circuitos em tempo polinomial.

B.4 Complexidade de Circuitos Quânticos

No modelo das máquinas de Turing, a complexidade computacional é especificada em termos do tempo ou do espaço que a máquina utiliza para realizar uma dada computação. Normalmente, a complexidade de tempo é mais freqüentemente adotada para descrever e comparar algoritmos, em termos de análises assintóticas do tamanho da entrada [Cormen et al. 2001].

No modelo dos circuitos quânticos, existem três medidas de complexidade possíveis, descritas a seguir:

1. Número total de portas: consiste na contagem no número total de portas utilizadas no circuito;

2. Profundidade do circuito: consiste na divisão do circuito em seqüências de instantes, em que a aplicação de uma porta requer um único instante t. A profundidade de um circuito, portanto, consiste no total de instantes t utilizados para realizar a computação. O número total de portas e a profundidade de um circuito podem ser diferentes, pois há situações em que uma ou mais portas, em um mesmo instante t, atuam sobre qubits distintos;

3. Largura do circuito: de acordo com esta medida de complexidade, também conhecida como espaço, contam-se o número de qubits no circuito.

Para ilustrar os conceitos apresentados, a Figura B.14 denota um circuito quântico que implementa o algoritmo de Deutsch, primeiro algoritmo quântico a ser proposto, capaz de determinar se uma função f é constante ou balanceada [Mermin 2007]. Este circuito possui um número total de 4 portas, profundidade igual a 3 e largura igual a 2.

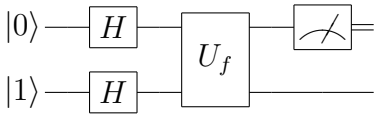

Figura B.14: Circuito quântico para o algoritmo de Deutsch.

Notas do Apêndice

Este capítulo apresentou a notação e os elementos dos circuitos quânticos, modelo computacional capaz de representar algoritmos da Computação Quântica.

De acordo com a notação de circuitos quânticos, as linhas horizontais representam a evolução do qubit ao longo do tempo e não é possível haver bifurcação nem retroalimentação. As operações sobre os qubits são denotadas por portas quânticas, de 1 ou múltiplos qubits, apresentando ou não controles. As portas implementam operações unitárias sobre qubits. De acordo com o Teorema Não-Clonagem a definição de portas que copiam estados quânticos não é permitida. Apesar disso, é possível construir estados emaranhados onde a cópia é habilitada até que não haja uma medição.

O modelo de circuitos quânticos é equivalente ao modelo da Máquina de Turing Quântica. Portanto, é possível expressar qualquer algoritmo quântico sob a forma de circuito, uma notação mais simples de ser assimilada, visto que possui uma representação visual e é similar aos circuitos lógicos clássicos. Existem métricas que definem a complexidade de circuitos, as quais levam em consideração o número de portas, a profundidade ou a largura do mesmo.

Apêndice C

Artigos Publicados

Quantum Permanent Compromise Attack to Blum-Micali Pseudorandom Generator

Elloá B. Guedes, Francisco M. de Assis, Bernardo Lula Jr.
IQuanta – Institute for Studies in Quantum Computation and Quantum Information
Federal University of Campina Grande
Rua Aprígio Veloso, 882 – Campina Grande – Paraíba – Brazil
elloaguedes@gmail.com, fmarcos@dee.ufcg.edu.br, lula@dsc.ufcg.edu.br

***Abstract*— This paper presents a quantum permanent compromise attack to the Blum-Micali pseudorandom generator whose security is based on the assumption of intractability of the discrete logarithm problem. The proposed attack makes use of the Grover's quantum search extension for multiple solutions and of quantum parallelism to recover the generator's internal state with high probability. This attack compromises the unpredictability of the Blum-Micali cryptographic secure pseudorandom generator, since it recovers all previous and future output, as well as the generator's seed. Compared to the classical equivalent attack, the quantum algorithm proposed has a quadratic speedup, and represents a menace against the security of a pseudorandom generator used in many real-world cryptosystems.**

Keywords*— *Blum-Micali Generator, Quantum Attack, Grover Algorithm

I. Introduction

Randomness is an essential requirement for any well-designed cryptographic application. Session keys, initialization vectors, salts to be hashed with passwords, unique parameters in digital signatures, and nonces in protocols are examples of randomness usage in cryptography [1]. However, random sources, like Geiger counts or radioactivity decay, are not available in all cryptographic systems. In face of this limitation, it is acceptable the use of an algorithmic alternative: the *pseudorandom numbers generators* (PRNGs).

The most useful type of pseudorandom processes updates a current sequence of numbers in a manner that appears to be random. Such a deterministic generator, f, yields numbers recursively, in a fixed sequence. The previous numbers (often just the single previous number) determine the next number:

$$x_i = f(x_{i-1}, \ldots, x_{i-k}). \tag{1}$$

Considering that the set of numbers directly representable in the computer is finite, the sequence will repeat. The set of values at the start of the recursion is called the *seed*. Each time the recursion is begun with the same seed, the same sequence is generated. A common requirement of PRNGs is that they possess good statistical properties, meaning their output approximates a sequence of true random numbers. Those properties can be assessed via statistical tests [2], [3].

An special type of PRNG with unpredictable outputs is the *cryptographically secure pseudorandom number generators* (CSPRNGs), i.e., given n consecutive output bits of a CSPRNG, there is no polynomial time algorithm that can predict the next bit with better than 50% chance of success [4]. Since randomnesss plays a major role in cryptographic systems, it is essential to analyze the vulnerability of PRNGs specially those that are used in real world cryptosystems, i.e., the CSPRNGs.

A family of CSPRNGs widely adopted in cryptography was stamped by Blum and Micali [4]. According to them, each generator from this construction must be composed by a function f, that is a permutation over a domain $\mathcal{D}$, and a binary function. Examples of generators founded on the Blum-Micali construction are Blum-Blum-Shub [5], Kaliski [6], and the Blum-Micali generator [4] whose security is based on the assumptions of intractability of the factoring, elliptic curve discrete logarithm and discrete logarithm problems, respectively.

However, *quantum computers* can implement efficient algorithms to certain problems where an efficient classical algorithm is not known. An example is the Shor's quantum factoring that may break the whole security of RSA algorithm when quantum computers become scalable [7]. Another relevant algorithm, the quantum search, was proposed by Grover [8] and performs a search in an unsorted N-sized database with $O(\sqrt{N})$ cost.

In this paper, it is proposed a quantum permanent compromise attack, based on the quantum search, to the Blum-Micali CSPRNG. Such attack retrieves the internal state of the generator and endanger all future and previous output, compromising the unpredictability. Moreover, this attack has a quadratic speedup to its classical counterpart.

The rest of this paper is organized as follows. In Section II the Blum-Micali generator is described and a simple exponential-order permanent compromise attack to this generator is introduced. In Section III the Grover algorithm is reviewed. In Section IV is introduced a quantum permanent compromise attack to the Blum-Micali pseudorandom generator. In the Section V related works are briefly discussed and in the Section VI conclusions and future work are presented.

II. Blum-Micali Generator

Let p be a large prime and $n = \lceil \log p \rceil$ the binary length of p. The set $\mathbb{Z}_p^* = \{1, 2, \ldots, p-1\}$ stands for the cyclic group under multiplication $\mod p$. Let g be a generator of $\mathbb{Z}_p^*$ and $x_0 \in_R \mathbb{Z}_p^*$. We denote $a \in_R A$ the random choice of an element a out of the set A.

The Blum-Micali generator (BM) is prepared with parameters (p,g,x_0) as previously described where x_0 is the seed of the generator, and p and g are the parameters publicly known. This generator produces pseudorandom bits using an one-way function $x_i = g^{x_{i-1}} \mod p$ over the domain $\mathbb{Z}_p^*$, and a hard-core predicate for the permutation, denoted by δ, as shown above:

$$x_i = g^{x_{i-1}} \mod p \qquad \text{exponential map} \tag{2}$$

$$b_i = \delta(x_i) \tag{3}$$

where δ is a binary function with the following definition:

$$\delta(x) = \begin{cases} 1 & \text{if } x > \frac{p-1}{2} \\ 0 & \text{otherwise.} \end{cases}$$

To illustrate the usage of BM, suppose a generator configured with parameters $(7, 3, 1)$, denoting p, g, and x_0, respectively. The diagram (4) shows the evolution of internal states and the bits outputted during this process.

$$\begin{array}{ccccccccc} 1 & \rightarrow & 3 & \rightarrow & 6 & \rightarrow & 1 & \rightarrow & \dots \\ \downarrow & & \downarrow & & \downarrow & & \downarrow & & \\ 0 & & 0 & & 1 & & 0 & & \end{array} \tag{4}$$

The security of the BM generator relies on the hardness of inverting the one-way function $x_{i+1} = g^{x_i} \bmod p$. This inversion is equivalent to solve the discrete logarithm problem and no efficient algorithm to perform this operation is known in classical computing [9]. In the following subsection we introduce a simple algorithm to attack to the Blum-Micali generator.

A. Permanent Compromise Attack to Blum-Micali Generator

A permanent compromise attack happens to a pseudo-random generator when its internal state is recovered and, in consequence, all previous and future output become predictable. This kind of attack compromises the unpredictability of the cryptographic secure PRNG, and also recovers the seed.

Some concepts are needed before the permanent compromise attack description. The BM generator's internal (unknown) state in time i is defined as an ordered set $X(i) \triangleq \{x_i, x_{i-1}, x_{i-2}, \dots, x_0\}$, where x_0 is the seed. Each $x_k \in X(i)$ is associated to a bit outputted by the generator, i.e., $b_k = \delta(x_k), k = 0, 1, \dots, i$. We call x_i the *representative* of the internal state $X(i)$.

Since the evaluation of the exponential in (2) is computationally efficient and considering that at some point the sequence will repeat, if any single element of a set $X(i)$ along its index are discovered then all the previous and future internal states can be recovered, i.e., all elements of $X(i)$ are found out.

To attack a BM generator used in a cryptosystem, an adversary has knowledge of the public parameters p and g, and has previously discovered a sequence of output bits ($\mathbf{b} \triangleq \{b_i, i = 1, 2, \dots\}$) produced by the generator under attack.

In the attempt to recover the internal state of the generator, the adversary needs to estimate an element $x_k \in X(i)$ along its index. Without loss of generality, consider $k = i$. Let $\hat{X}_i$ be the set of guesses to $x_i \in X(i)$. The set $\hat{X}_i$ will be referred as the *estimator set relative to* x_i, or plainly *estimator set*. For example, as stated in the BM initialization, the seed x_0 is randomly chosen from $\mathbb{Z}_p^*$, so the adversary would start in the attempt to attack the generator with the estimator set $\hat{X}_0 = \mathbb{Z}_p^*$.

With a given estimator set $\hat{X}_i$ that contains x_i and the knowledge of the bit b_{i+1} outputted by the generator, the adversary would proceed as follows to produce $\hat{X}_{i+1}$. Compute $A_{i+1} = g^{\hat{X}_i} \bmod p$, i.e., the image of $\hat{X}_i$ under action of the map $x \rightarrow g^x \bmod p$. Let $A_{i+1} = A_{i+1}^{(0)} \cup A_{i+1}^{(1)}$ be the partition of A_{i+1} due to the BM rule, that is, $x \in A_{i+1}^{(0)}$ iff $\delta(x) = 0$ and similar to $A_{i+1}^{(1)}$. The estimator set $\hat{X}_{i+1}$ will be given by:

$$\hat{X}_{i+1} = A_{i+1}^{(b_{i+1})}. \tag{5}$$

Notice that $\hat{X}_{i+1},\ i = 0, 1, 2, \dots$ only contains elements out of the class defined by $A_{i+1}^{(b_{i+1})}$. The Algorithm 1 synthesizes the procedure followed by the adversary.

Algorithm 1 Pseudocode of the algorithm used by the adversary to promove a permanent compromise attack to a BM generator. In this algorithm, δ denotes the function shown in (3); j is the number of bits in **b**; and every set $\hat{X}_i$ used for the first time is an empty set.

```
i ← 1
X̂_0 ← {1, 2, ..., p − 1}
while (i ≤ j) do
  for all x ∈ X̂_{i−1} do
    if δ(g^x mod p) = b_i then
      X̂_i ← X̂_i ∪ {g^x mod p}
    end if
  end for
  if i ≠ j then
    i ← i + 1
  end if
end while
print X̂_i
```

To exemplify the described permanent compromise state attack, suppose that the adversary eavesdropped 2 sequenced output bits, $\mathbf{b} = \{10\}$, from a BM with parameters $p = 7$ and $g = 3$. He would start his estimators to the seed with the set $\hat{X}_0 = \{1, 2, \dots, 6\}$. With the information that the first bit observed was $b_1 = 1$, he would discard the numbers lesser than $(7 - 1)/2 = 3$ and perform the operation $g^x \bmod p$ in the remaining ones, resulting in the set of estimators to x_1, $\hat{X}_1 = \{4, 5, 6\}$. Proceeding the same way, but with the next observed bit $b_2 = 0$, the intruder would update from $\hat{X}_1$ his estimators to x_2, resulting in $\hat{X}_2 = \{1\}$. In this case, with only two intercepted bits, the adversary could retrieve, with 100% of sure, the internal state of the generator.

B. Correctness and Analysis of the Algorithm (1)

To verify that the permanent compromise attack to BM recovers the internal state, it must be proved that: (i) every estimator set $\hat{X}_i$ contains $x_i \in X(i)$; and (ii) given sufficient bits in **b**, the size of the set $\hat{X}_i$ is unitary for i large enough or the adversary will easily to predict the next generator's output. The skecth of the proofs to the requirements (i) and (ii) are presented below. In order to prevent trivialities we assume that x_0 is member of a larger cycle in the functional graph [10] of the exponential map (2). This assumption is clearly acceptable since short cycles correspond to high predictability of the sequence.

Proof of (i) *– Every set* $\hat{X}_i$ *of estimators contains the correspondent representative* $x_i \in X(i)$*:* Induction in i. Clearly, the claim is valid for $i = 0$ as $x_0 \in \hat{X}_0 = Z_p^*$. Assume $x_i \in \hat{X}_i$ (induction hypothesis). It must be shown that $x_{i+1} \in \hat{X}_{i+1}$. Indeed, from (5), $y \in \hat{X}_{i+1}$ if only if (1) $y = g^x \bmod p$ for some $x \in \hat{X}_i$ and, (2) $\delta(y) = b_{i+1}$. But,

by the induction hypothesis, $x_i \in \hat{X}_i$ and, according to BM productions, $\delta(x_{i+1} = g^{x_i} \bmod p) = b_{i+1}$ so x_{i+1} attains both requirements (1) and (2). We have done. ■

Proof of (ii) (sketch) – Cardinality $|\hat{X}_i| = 1$ for i enough large or the next bit is fully predictable: In the $i-$th step, prior observing b_i, the adversary calculate $A_i = g^{\hat{X}_{i-1}} \bmod p = A_i^{(0)} \cup A_i^{(1)}$. Define $q = |\hat{X}_i|/|\hat{X}_{i-1}|$, then observe that $q = \Pr[b_i = 0|\hat{X}_{i-1}]$ or $q = \Pr[b_i = 1|\hat{X}_{i-1}]$ due to the BM rule. Then the uncertainty of b_i is $\mathcal{H}(q)$ where $\mathcal{H}(q)$ stands for the binary entropy calculated in q. Note that if $A_i^{(0)} = \emptyset$ then $q = 1$ and b_i is completely predictable and similar is true if $A_i^{(1)} = \emptyset$. Thus we assume that a good choice of parameters x_0, g and p should hold $\mathcal{H}(q) > 1 - \epsilon$, where ϵ is a small positive number. Therefore $q = |\hat{X}_i|/|\hat{X}_{i-1}|$ is near enough to $1/2$ or the bit b_i is fully predictable. This conclude the sketch of the proof of (ii). ■

To analyze the complexity of the permanent compromise attack described it is important not forget that p in practical applications is a large prime, i.e., $p \approx 2^n$, and that evaluations of (2) have unitary cost of the exponent. In the adversary's algorithm the modular exponentiation is evaluated to modular residues of p, thus the complexity is $O(p) \approx O(2^n)$ to a classical computer.

III. Grover's Quantum Search Algorithm

The Grover's quantum search algorithm performs a generic search for a solution in an unsorted database [8]. This algorithm is adequate to many problems where the best-known algorithm is to naively search through the potential solutions until one is found.

Suppose that an unsorted database contains N elements and M solutions. To represent the N elements of the database in binary, there are necessary $n = \lceil \log N \rceil$ bits, i.e., n is the problem input size. To perform the Grover's search in this database the steps described below must be followed:

1) Preparation of two registers: the first containing n qubits initialized as $|0\rangle$, and the second register containing a single $|1\rangle$ *ancilla* qubit;
2) Hadamard transformation: must be applied in both registers, producing $\frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle$ and $|-\rangle$;
3) Grover iteration: Repeat k times the following steps:
   a) Phase inversion: Invert the phase of the states in the first register that are solutions to the search problem. Use the second register as an auxiliary to this operation;
   b) Amplitude amplification: Re-invert the phases that were inverted in the previous step over the total average. Due to the fact that a quantum system is described by an unitary vector, this process will increase the amplitudes of the solutions and decreases the others;
4) Measurement: Perform a measurement in the first register.

The step 2 consisted in the Hadamard transformation of the input, and thus the state before the Grover's iteration can be written considering the solutions (subspace $|\psi_{good}\rangle$) and the non-solutions (subspace $|\psi_{bad}\rangle$):

$$
\begin{aligned}
|\psi\rangle &= H^{\otimes n+1} |0\rangle^{\otimes n} |1\rangle && (6)\\
&= \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} |k\rangle && (7)\\
&= \sqrt{p_{good}} |\psi_{good}\rangle + \sqrt{p_{bad}} |\psi_{bad}\rangle && (8)\\
&= \sin(\theta) |\psi_{good}\rangle + \cos(\theta) |\psi_{bad}\rangle && (9)
\end{aligned}
$$

where $p_{good} = \sum_{x \in X_{good}} |\alpha_x|^2$, $\alpha_x = \frac{1}{\sqrt{2^n}}$, $p_{bad} = 1 - p_{good}$, $\theta \in \left(0, \frac{\pi}{2}\right)$, and $\sin^2(\theta) = p_{good}$.

The operator G, denoting the Grover iteration, must be applied $k \approx \frac{\pi}{4}\sqrt{\frac{N}{M}}$ times in order to amplify the amplitudes of the solutions to the problem:

$$G^k |\psi\rangle = \cos((2k+1)\theta) |\psi_{bad}\rangle + \sin((2k+1)\theta) |\psi_{good}\rangle . \quad (10)$$

The Grover's quantum search algorithm provides a quadratic speed-up over its best-known classical counterpart. The extremely wide applicability of searching problems makes this algorithm interesting and important. The next section shows an application of the Grover's algorithm to endanger the security of a BM generator.

IV. Quantum Permanent Compromise Attack

The quantum algorithm that performs the permanent compromise attack to Blum-Micali is composed of two main parts: the first part, illustrated in the circuit of Figure 1, that is responsible for the identification of the estimator set to the representative of $X(i)$; and the second part, that follows the first and is illustrated in the Figure 2, that performs the amplitude amplification with the Grover's quantum search algorithm.

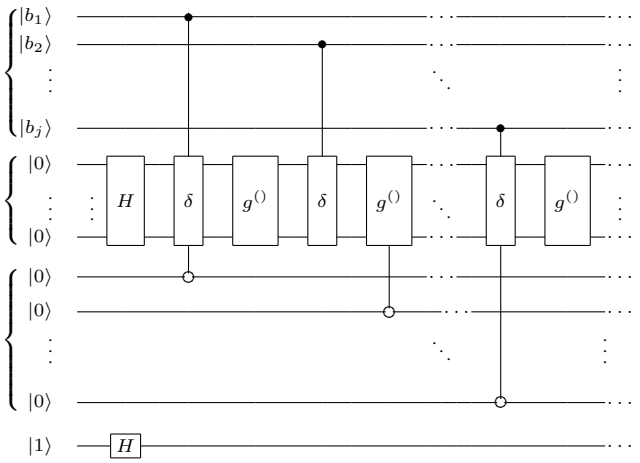

Figure 1. First part of the quantum circuit that implements the quantum permanent compromise attack to Blum-Micali pseudorandom generator.

This algorithm starts creating a superposition of qubits, representing all the elements in $\hat{X}_0 = \mathbb{Z}_p^*$, and uses the information of the **b** bits discovered to mark which of these elements could be in the internal state $X(i)$. After this phase, the Grover algorithm is run and the amplitude of

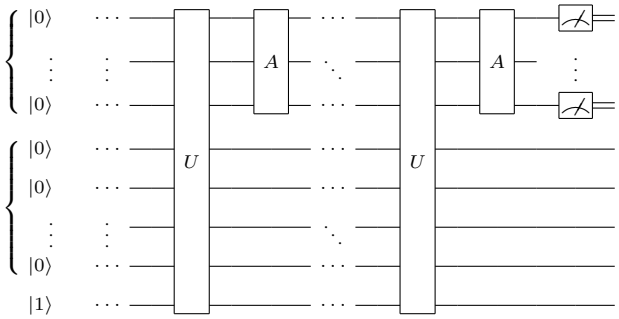

Figure 2. Second part of the quantum circuit that implements the quantum permanent compromise attack to Blum-Micali pseudorandom generator. This part follows the first one and implements the Grover's quantum search.

the elements marked as solutions are amplified, being able to be measured with high probability. The input for the circuit that implements the quantum permanent compromised is described as follows:

1) The first register is composed by the sequence of bits discovered by the intruder simply codified as qubits;
2) The second register, containing $\lceil \log p \rceil$ qubits, are used to represent the elements of $\hat{X}_i$;
3) Containing as much qubits as the first register, this third register is composed of *ancilla* qubits, that will help in the identification of the estimators for $x \in X$;
4) The fourth register contains the auxiliary qubit for the Grover's quantum search algorithm.

The gates δ and $g^{()}$ in the first part, implement the following operations:

$$\delta_c \left|x\right\rangle \left|0\right\rangle = \begin{cases} \left|x\right\rangle \left|0\right\rangle & \text{if } x \notin \mathbb{Z}_p^* \\ \left|x\right\rangle \left|1\right\rangle & \text{if } x > \frac{(p-1)}{2} \quad \text{and } c = 1 \\ \left|x\right\rangle \left|1\right\rangle & \text{if } x \leq \frac{(p-1)}{2} \quad \text{and } c = 0 \\ \left|x\right\rangle \left|0\right\rangle & \text{otherwise} \end{cases}$$

$$g^{()} \left|x\right\rangle = \begin{cases} x & \text{if } x \notin \mathbb{Z}_p^* \\ g^x \bmod p & \text{otherwise} \end{cases}$$

Both gates are reversible and can be implement efficiently in a quantum computer, since they are already implemented efficiently on a classical computer. After this first part, given j bits in $\mathbf{b}$, it is expected that the estimators to x_j will be the states associated to $\left|11\ldots1\right\rangle$ in the third register. The gates U and A of the quantum searching part denotes, respectively, the phase inversion and amplitude amplification of Grover's algorithm. The U oracle implements the function $f(x) = 1$ if the third register is equal to $\left|11\ldots11\right\rangle$, and $f(x) = 0$ otherwise. The applications of U and A are equivalent to the Grover iterations from (10).

In the attempt to exemplify the quantum permanent compromise attack, suppose that an intruder recovered the bits $\mathbf{b} = \{001\}$ and knows that $p = 7$ and $g = 3$ for a certain BM. He also possess a quantum computer that implements the quantum permanent compromise attack algorithm described, and wants to use this algorithm to recover the internal state of the generator. The adversary would prepare the first register with the state $\left|001\right\rangle$; the second register with 3 qubits initialized with $\left|0\right\rangle$; the *ancilla* register with the state $\left|000\right\rangle$; and the Grover auxiliary *ancilla* qubit with $\left|1\right\rangle$, resulting in the following initial state:

$$\left|\psi_0\right\rangle = \left|001\right\rangle \left|000\right\rangle \left|000\right\rangle \left|1\right\rangle \tag{11}$$

Since the first register is just used to control the application of δ, it will be omitted in the following steps. According to the circuit in the Figure 1, a Hadamard gate must be applied in the second and fourth registers, and the other registers must be left unchanged, resulting:

$$\begin{aligned} \left|\psi_1\right\rangle &= H^{\otimes 3} \otimes \mathbb{1}^{\otimes 3} \otimes H \left[\left|000\right\rangle \left|000\right\rangle \left|1\right\rangle\right] && (12) \\ &= \frac{1}{\sqrt{2^3}} \sum_{k=0}^{2^3-1} \left|k\right\rangle \left|000\right\rangle \left|-\right\rangle && (13) \\ &= \frac{1}{\sqrt{8}} \left(\left|0\right\rangle + \left|1\right\rangle + \ldots + \left|7\right\rangle\right) \left|000\right\rangle \left|-\right\rangle && (14) \end{aligned}$$

The first application of δ is controlled by a 0 qubit and targets the first *ancilla* qubit to 1 if the second register is lesser or equal than $(p-1)/2$. As consequence of δ application, the $\left|\psi_1\right\rangle$ state evolves to $\left|\psi_2\right\rangle$:

$$\begin{aligned} \left|\psi_2\right\rangle &= \delta_0 \left|\psi_1\right\rangle && (15) \\ &= \frac{1}{\sqrt{8}} \left[\left(\left|0\right\rangle + \left|4\right\rangle + \ldots + \left|7\right\rangle\right) \left|000\right\rangle + \right. \\ &+ \left. \left(\left|1\right\rangle + \left|2\right\rangle + \left|3\right\rangle\right) \left|100\right\rangle\right] \left|-\right\rangle && (16) \end{aligned}$$

The next step is to perform the transformation $g()$:

$$\begin{aligned} \left|\psi_3\right\rangle &= g^{()} \left|\psi_2\right\rangle && (17) \\ &= \frac{1}{\sqrt{8}} \left[\left(\left|0\right\rangle + \left|4\right\rangle + \left|5\right\rangle + \left|1\right\rangle + \left|7\right\rangle\right) \left|000\right\rangle + \right. \\ &+ \left. \left(\left|3\right\rangle + \left|2\right\rangle + \left|6\right\rangle\right) \left|100\right\rangle\right] \left|-\right\rangle && (18) \end{aligned}$$

The expression of the state $\left|\psi_3\right\rangle$ tells that $\hat{X}_1 = \{3, 2, 6\}$. But, the intruder knows two more bits, and also the estimators set $\hat{X}_2$ and $\hat{X}_3$ can be obtained.

The application of δ_0 controlled by the second qubit of the first register results:

$$\begin{aligned} \left|\psi_4\right\rangle &= \frac{1}{\sqrt{8}} \left[\left(\left|0\right\rangle + \left|4\right\rangle + \left|5\right\rangle + \left|7\right\rangle\right) \left|000\right\rangle + \right. \\ &+ \left. \left|1\right\rangle \left|010\right\rangle + \left|6\right\rangle \left|100\right\rangle + \left(\left|3\right\rangle + \left|2\right\rangle\right) \left|110\right\rangle\right] \left|-\right\rangle && (19) \end{aligned}$$

The result of $g^{()}$ to $\left|\psi_5\right\rangle$ is given below:

$$\begin{aligned} \left|\psi_5\right\rangle &= g^{()} \left|\psi_4\right\rangle && (20) \\ &= \frac{1}{\sqrt{8}} \left[\left(\left|0\right\rangle + \left|4\right\rangle + \left|5\right\rangle + \left|7\right\rangle\right) \left|000\right\rangle + \right. \\ &+ \left. \left|3\right\rangle \left|010\right\rangle + \left|1\right\rangle \left|100\right\rangle + \left(\left|6\right\rangle + \left|2\right\rangle\right) \left|110\right\rangle\right] \left|-\right\rangle && (21) \end{aligned}$$

To conclude the first phase, the δ gate, controlled by the third qubit of the first register and targeting the third qubit of the third register, needs to be applied to the state $\left|\psi_5\right\rangle$:

$$\begin{aligned} \left|\psi_6\right\rangle &= \delta_1 \left|\psi_5\right\rangle && (22) \\ &= \frac{1}{\sqrt{8}} \left[\left(\left|0\right\rangle + \left|7\right\rangle\right) \left|000\right\rangle + \left(\left|4\right\rangle + \left|5\right\rangle\right) \left|001\right\rangle \right. \\ &+ \left. \left|3\right\rangle \left|010\right\rangle + \left|1\right\rangle \left|100\right\rangle + \left|2\right\rangle \left|110\right\rangle + \left|6\right\rangle \left|111\right\rangle\right] \left|-\right\rangle && (23) \end{aligned}$$

After the execution of the first phase of the algorithm, the state $|6\rangle$ in Eq. (23) is the only one associated to $|111\rangle$, identifying that the set $\hat{X}_3$ is unitary and, therefore, $x_3 = 6$. Although the representative is already identified, this information is accessible only at the quantum level – at the final of the first phase a measurement would return any number from 0 to 7 with the same probability. Therefore, it is needed to perform Grover iterations, the second phase of the algorithm, to increase the probability that the representative will be returned after a measurement.

To simplify the calculus, the state $|\psi_6\rangle$ can be rewritten in the following manner:

$$|\psi_6\rangle = \sqrt{\frac{7}{8}}\left|\neg\hat{X}_i\right\rangle|y\rangle|-\rangle + \frac{1}{\sqrt{8}}\left|\hat{X}_i\right\rangle|111\rangle|-\rangle \quad (24)$$

$$= \sin(\theta)\left|\psi_{\hat{X}_i}\right\rangle + \cos(\theta)\left|\psi_{\neg\hat{X}_i}\right\rangle \quad (25)$$

where $i = 3$; $\left|\hat{X}_i\right\rangle = |6\rangle$; $\left|\neg\hat{X}_i\right\rangle$ denotes all the states that aren't in the estimator set $\hat{X}_3$; $y \neq 111$; $\left|\psi_{\hat{X}_i}\right\rangle = |6\rangle|111\rangle|-\rangle$; $|\psi_{\neg\hat{i}}\rangle = \left|\neg\hat{X}_i\right\rangle|y\rangle|-\rangle$; $\theta \in (0, \frac{\pi}{2})$ satisfies $\sin^2(\theta) = \frac{1}{8}$; and, thus $\theta \approx 0.36$ radians.

The optimal number k of iterations to be performed is $\frac{\pi}{4}\sqrt{\frac{8}{1}} \approx 2$. So, two Grover iterations on $|\psi_6\rangle$ results:

$$|\psi_7\rangle = G^2|\psi_6\rangle \quad (26)$$

$$= \cos[(2\cdot 2+1)\theta]\left|\psi_{\bar{i}}\right\rangle + \sin[(2\cdot 2+1)\theta]|\psi_i\rangle \quad (27)$$

$$= \cos(5\theta)\left|\psi_{\bar{i}}\right\rangle + \sin(5\theta)|\psi_i\rangle \quad (28)$$

$$= \cos(1.8)\left|\psi_{\bar{i}}\right\rangle + \sin(1.8)|\psi_i\rangle \quad (29)$$

This result states that a measurement in the second register of $|\psi_7\rangle$ will return the representative to $X(3)$ ($x_3 = 6$) with probability $|\sin(1.8)|^2 = 94,53\%$ of sure. With this information the intruder will be able to retrieve all the set $X(i)$ of internal states from the generator under attack, endangering its unpredictability.

The quantum permanent compromise attack to BM described in this section has total complexity equal to the sum of the complexities of each phase. In the first one, the cost is equal to $O(1)$, because of $2\cdot j\cdot O(1)$ oracle's operations performed; the second part has total cost equal to the Grover's algorithm, $O(\sqrt{p}) = O(\sqrt{2^n})$ because $p \approx 2^n$. So, the total cost of the quantum algorithm is $O(1)+O(\sqrt{2^n}) = O(\sqrt{2^n})$. This result represents a quadratic speedup over its classical counterpart.

V. Related Work

Classical attacks to pseudorandom generators have been studied for many researchers, not only with cryptographic purposes. Kelsey et al. [1] proposed a taxonomy that classifies attacks to PRNGs in six different categories and also discuss their extensions. According to these authors, the study of cryptanalytic attacks on PRNGs has practical and theoretical applications because (i) there aren't any widespread understanding of the possible attacks to PRNGS; (ii) PRNGs are single point of failure in many real-world cryptosystems; and, (iii) many systems use badly-designed PRNGs or use them in ways to make various attacks easier that they need to be.

So far, regarding quantum attacks to pseudorandom generators, no references were found in the literature that present such attacks. Since the proposed algorithm characterizes a speedup over its classical counterpart, it opens up the possibility to use quantum computation to endanger the security of such generators.

VI. Conclusions and Future Work

This paper introduced a quantum permanent compromise attack to the Blum-Micali generator. This algorithm uses quantum searching with multiple solutions and parallelism to retrieve the generator's internal state with high probability. The algorithmic complexity is $O(\sqrt{2^n})$ contrasting with $O(2^n)$ of the best classical algorithm known, where n is the input size.

To the authors's knowledge, the attack introduced in this work is the first proposal of quantum attacks against pseudorandom numbers generators. Furthermore, this attack brings results against the security of a generator used in real-world cryptographic applications.

In future works, the authors intend to analyze the adoption of some elements from the quantum algorithm to the discrete logarithm problem into the permanent compromise attack.

Acknowledgements

The authors gratefully acknowledge the financial support rendered by the Brazilian National Council for the Improvement of Higher Education (CAPES), and the useful discussions with Francine Melo and Gilson O. Santos.

References

[1] J. Kelsey, B. Schneider, D. Wagner, and C. Hall, "Cryptanalytic attacks on pseudorandom number generators," *Lecture Notes in Computer Science*, vol. 1372/1998, pp. 168–188, 1998.

[2] A. Rukhin, "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," National Institute of Standards and Technology, Tech. Rep., 2008.

[3] G. Marsaglia, "The Marsaglia Random Number CDROM, including the DIEHARD Battery of Tests of Randomness," 1995.

[4] M. Blum and S. Micali, "How to generate cryptographically strong sequences of pseudorandom bits," *SIAM J. Comput.*, vol. 13 (4), pp. 850–864, 1984.

[5] L. Blum, M. Blum, and M. Shub, "A simple unpredictable pseudo-random number generator," *SIAM Journal on Computing*, vol. 15, pp. 364–383, 1986.

[6] B. S. Kaliski, "Elliptic curves and cryptography: A pseudorandom bit generator and other tools," Ph.D. dissertation, MIT, Cambridge, MA, USA, 1988.

[7] P. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, pp. 1484–1509, 1997.

[8] L. K. Grover, "Quantum mechanics helps in searching for a needle in a haystack," *Physical Review Letter*, vol. 79, pp. 325–328, 1997.

[9] A. Sidorenko and B. Schoenmakers, "State recovery attacks on pseudorandom generators," in *Western European Workshop on Research in Cryptology*, 2005, pp. 53–63.

[10] D. Cloutier, "Mapping the discrete logarithm," Senior thesis – Rose-Hulman Institute of Technology, 2005.

A Generalized Quantum Permanent Compromise Attack to the Blum-Micali Construction

Elloá B. Guedes, Francisco M. de Assis, Bernardo Lula Jr.

[1]IQuanta – Institute for Studies in Quantum Computation and Quantum Information
Federal University of Campina Grande
Rua Aprígio Veloso, 882 – Campina Grande – Paraíba – Brazil

elloaguedes@gmail.com, fmarcos@dee.ufcg.edu.br, lula@dsc.ufcg.edu.br

Abstract. *The Blum-Micali construction defines cryptographically secure pseudorandom generators widely adopted in many real-world cryptosystems. By endangering this construction, the security of certain cryptographic applications may become vulnerable. In the attempt to menace the security of Blum-Micali construction, this paper presents a generalized quantum permanent compromise attack to it. This attack is based on Grover's quantum search and on quantum parallelism to retrieve the generator's internal state. Experimental results are also provided to support the number of bits the adversary must know to perform the attack successfully against the Blum-Micali generator. The attack may bring new security requirements to be imposed on pseudorandom numbers generators.*

1. Introduction

There is a close relationship between cryptography and randomness. The security of cryptographic algorithms usually depends on the random choice of keys and bit sequences. Ideally, this random data should be originated from a non-deterministic phenomena, i.e., whose behavior cannot be predicted.

However, digital computers cannot generate random numbers, and it is generally not convenient to connect a computer to some external source of random events. It is possible to overcome this disadvantage if there is some source of *pseudorandom numbers* – algorithms that yield numbers that appears to be random. Many methods have been suggested in the literature for generating such pseudorandom numbers [Gentle 2003].

A *pseudorandom numbers generator* (PRNG) is a function f that yields numbers recursively, in a fixed sequence. The previous numbers (often just the single previous number) determine the next number:

$$x_i = f(x_{i-1}, \ldots, x_{i-k}) \tag{1}$$

Considering that the set of numbers directly representable in the computer is finite, the sequence will repeat. The set of values at the start of the recursion is called the *seed*. Each time the recursion is begun with the same seed, the same sequence is generated. A common requirement of PRNGs is that they possess good statistical properties, meaning their output approximates a sequence of true random numbers. Those properties can be assessed via statistical tests [Rukhin 2008, Marsaglia 1995].

In cryptography, the requirements for randomness are more stringent than in other domains: the known conditional probability of the next random number occur, given the previous history, is no different from the known unconditional probability

[Blum and Micali 1984]. This requirement defines a special class of pseudorandom generators – the *cryptographically secure pseudorandom number generators* (CSPRNGs).

These generators can be derived from one-way permutations with hard-core predicates as defined by the *Blum-Micali construction*. Examples of generators founded on the Blum-Micali construction are Blum-Blum-Shub [Blum et al. 1986], Kaliski [Kaliski 1988], and the Blum-Micali generator [Blum and Micali 1984] whose security is based on the assumptions of intractability of the factoring, elliptic curve discrete logarithm and discrete logarithm problems, respectively. Since attacks focus on the parts of the cryptosystems that are most susceptible, it is essencial to analyze the vulnerability of such generators against menaces.

Quantum computing, a computational paradigm based on Quantum Physics, has been proposing efficient algorithms to certain problems where an efficient classical algorithm is not known. One of the examples is related to the security of the RSA, that is based on the assumption of hardness of factoring to a classical computer. However, Shor proposed a quantum factoring algorithm capable to solve this problem efficiently [Shor 1997]. This result opened up the possibility to endanger the security of classical cryptosystems with this computational paradigm.

Towards attacks to pseudorandom numbers generators with quantum computing, Guedes et al. [Guedes et al. 2010b] proposed a permanent compromise attack to the Blum-Micali generator. This attack was based on the Grover algorithm, a quantum procedure to perform search in an unsorted database [Grover 1997]. According to the authors, the proposed attack has a quadratic speedup over its classical counterpart.

This paper intends to generalize the quantum permanent compromise attack proposed by Guedes et al. to the whole Blum-Micali construction, showing how the quantum gates can be extended according to each generator. Furthermore, this paper presents an experimental study to estimate the number of bits the adversary needs to know to perform the attack successfully against the Blum-Micali generator. The results obtained increase the confidence in boundaries previously estimated.

The rest of this paper is organized as follows. The Section 2 presents the concepts of the Blum-Micali construction, the permanent compromise attack to it, and also an experimental study regarding attacks to the Blum-Micali generator. In Section 3 the two phases and number of Grover's iterations from the generalized quantum attack to the Blum-Micali construction are characterized, an example is presented to illustrate such attack. In the Section 4 related works are briefly discussed and in the Section 5 conclusions and future work are drawn.

2. The Blum-Micali Construction

The Blum-Micali construction is a family of pseudorandom numbers generators widely adopted in cryptography [Blum and Micali 1984]. The generators from this family are defined by a one-way permutation ρ over a domain $\mathcal{D}$ and a hard-core predicate ϕ for the one-way permutation. The seed x_0 is obtained through a random choice of an element in the domain, denoted by $x_0 \in_R \mathcal{D}$. The productions (bits) from these generators are obtained as follows:

$$x_i = \rho(x_{i-1}) \quad (2)$$

$$b_i = \phi(x_i) \quad (3)$$

Three generators are the main representatives of this family: the Blum-Blum-Shub (BBS), Kaliski, and Blum-Micali (BM) generators.

The Blum-Blum-Shub generator's one-way permutation is based on quadratic residues modulo M, where M is the product of two primes both congruent to $3 \bmod 4$. Its ϕ function is characterized by the simple extraction of the j-th bit, where j is determined a priori. The security of the BBS generator is based on the assumption of the hardness of factoring [Blum et al. 1986].

The Kaliski generator is founded on the elliptic curve discrete logarithm problem. Its seed is a random point at the curve and the hard-core predicate for the one-way permutation is a binary test to the x abscissa, comparing it to the value of a large prime previously fixed [Kaliski 1988, Sidorenko and Schoenmakers 2005].

The Blum-Micali generator is based on modular exponentiation. This generator will be used to illustrate the examples and concepts along this paper. For this reason, its characteristics will be presented in details in the next section. After that, a permanent compromise attack to the Blum-Micali construction will be presented.

2.1. The Blum-Micali Generator

Let p be a large prime and $n = \lceil \log p \rceil$ the binary length of p. The set $\mathbb{Z}_p^* = \{1, 2, \ldots, p-1\}$ stands for the cyclic group under multiplication $\bmod p$. Let g be a generator of $\mathbb{Z}_p^*$.

The Blum-Micali generator is prepared with parameters (p,g,x_0) as previously described where x_0 is the seed of the generator, and p and g are the parameters publicly known. This generator produces pseudorandom bits using the one-way function $x_i = g^{x_{i-1}} \bmod p$ over the domain $\mathbb{Z}_p^*$, and a hard-core predicate for the permutation, denoted by δ, as shown below:

$$x_i = g^{x_{i-1}} \bmod p \tag{4}$$
$$b_i = \delta(x_i) \tag{5}$$

where δ is a binary function with the following definition:

$$\delta(x) = \begin{cases} 1 & \text{if } x > \frac{p-1}{2} \\ 0 & \text{otherwise.} \end{cases}$$

The security of the BM generator relies on the hardness to solve the inverse of the one-way function (4). This inversion is equivalent to solve the discrete logarithm problem and no efficient algorithm to perform this operation is known in classical computing.

To illustrate how the bits are outputted by the BM generator, suppose that it was initialized with parameters ($p = 7$, $g = 5$, $x_0 = 6$). The bits produced are shown in the Diagram (6).

$$\begin{array}{ccccccccc} 6 & \rightarrow & 1 & \rightarrow & 5 & \rightarrow & 3 & \rightarrow & \ldots \\ \downarrow & & \downarrow & & \downarrow & & \downarrow & & \\ 1 & & 0 & & 1 & & 0 & & \end{array} \tag{6}$$

2.2. Permanent Compromise Attack to the Blum-Micali Construction

A permanent compromise attack happens to a pseudorandom generator when its internal state is recovered and, in consequence, all previous and future output become predictable. This kind of attack compromises the unpredictability of the CSPRNG, and also recovers the seed.

Some concepts are needed before the permanent compromise attack description. A generator's internal (unknown) state in time i is defined as an ordered set $X(i) \triangleq \{x_i, x_{i-1}, x_{i-2}, \ldots, x_0\}$, where x_0 is the seed. Each $x_k \in X(i)$ is associated to a bit outputted by the generator, i.e., $b_k = \phi(x_k), k = 0, 1, \ldots, i$. Call x_i the *representative* of the internal state $X(i)$.

Since the evaluation of the one-way permutation ρ is computationally efficient and considering that at some point the sequence will repeat, if any single element of a set $X(i)$ along its index are discovered then all the previous and future internal states can be recovered, i.e., all elements of $X(i)$ are found out.

To attack one of the generator from the BM construction, an adversary has knowledge (i) of the type of the generator, (ii) of the public parameters, and (iii) has previously discovered a sequence of output bits ($\mathbf{b} \triangleq \{b_i, i = 1, 2, \ldots\}$) produced by the generator under attack.

In the attempt to recover the internal state of the generator, the adversary needs to estimate an element $x_k \in X(i)$ along its index. Without loss of generality, consider $k = i$. Let $\hat{X}_i$ be the set of guesses to $x_i \in X(i)$. The set $\hat{X}_i$ will be referred as the *estimator set relative to* x_i, or plainly *estimator set*. For example, as stated previously, the seed x_0 is randomly chosen from $\mathcal{D}$, so the adversary would start in the attempt to attack the generator with the estimator set $\hat{X}_0 = \mathcal{D}$.

With a given estimator set $\hat{X}_i$ that contains x_i and the knowledge of the bit b_{i+1} outputted by the generator, the adversary would proceed as follows to produce $\hat{X}_{i+1}$. Compute $A_{i+1} = \rho(\hat{X}_i)$, i.e., the image of $\hat{X}_i$ under action of the map $x \mapsto \rho(x)$.

Let $A_{i+1} = A_{i+1}^{(0)} \cup A_{i+1}^{(1)}$ be the partition of A_{i+1} due to the ϕ rule, that is, $x \in A_{i+1}^{(0)}$ iff $\phi(x) = 0$ and similar to $A_{i+1}^{(1)}$. The estimator set $\hat{X}_{i+1}$ will be given by:

$$\hat{X}_{i+1} = A_{i+1}^{(b_{i+1})}. \tag{7}$$

Notice that $\hat{X}_{i+1},\ i = 0, 1, 2, \ldots$ only contains elements out of the class defined by $A_{i+1}^{(b_{i+1})}$. The proofs of correctness of this algorithm can be seen in the paper of Guedes et al. [Guedes et al. 2010b]. These authors also suggest boundaries to the number of bits the adversary must know to perform the attack successfully, stating that this number is bounded by $\log p$ to the BM generator. The next section will present an experimental study performed in order to verify if practical results would follow this theoretical predicition

2.3. Experimental Study

This section presents an experimental study with the BM generator. This experiment can be explained as a game played between the adversary and the generator. To each generator

configured with parameters (p, g,x_0), the adversary would ask for many bits as necessary until he could predict the generator's next output with 100% of sure.

The parameters (p, g,x_0) to initialize each generator were configured as follows:

1. The generator g: four values of g where chosen (3, 5, 17, and 19). The requirement for different values of g is explained because if a single value were chosen it could bias the number of bits resultant;
2. The prime p: all prime values in the range $[257, 17863]$ were up to be used, however only those where $p - 1$ and g were coprimes were considered;
3. The seed x_0: uniformly sampled in the range $[1, p - 1]$ as suggested by the initialization of the BM generator.

After each game played between a generator and an adversary, the number of bits demanded was stored along with the value of p. This number of bits asked by the adversary until he could predict the generator's output with a 100% sure is the response variable of the experiment. It should be noticed that situations where the generator contained a fixed point in the discrete logarithm functional graph were not considered [Cloutier 2005].

The null hypothesis (H_0) of the experiment is that the number of bits required by the adversary is bounded by $\log p$, and the alternative hypothesis (H_a) is that this number is not bounded by $\log p$, where p is a large prime and one of the parameters to the BM generator. The level of confidence chosen was 95%[1]. A modified zero mean will be performed to verify the hypothesis under test.

The results of the attacks were summarized taking into account the number of bits in p, defining 6 groups with values of p from 8 to 14 bits. These results can be seen in the Table 1.

Table 1. Experimental results grouped according to the number of bits in p.

Bits in p	Required Sample Size	Number of Samples	Mean	Standard Deviation	Median	Confidence Interval
8	168	561	$8,044$	$1,333$	$8,000$	$(7,933, 8,155)$
9	351	453	$9,110$	$2,178$	$9,000$	$(8,909, 9,311)$
10	329	354	$10,107$	$2,341$	$10,000$	$(9,862, 10,3520)$
11	297	591	$11,090$	$2,441$	$11,000$	$(10,892, 11,286)$
12	241	1074	$12,011$	$2,379$	$12,000$	$(11,863, 12,158)$
13	196	1979	$13,111$	$2,533$	$13,000$	$(12,999, 13,222)$
14	183	1679	$14,008$	$2,420$	$14,000$	$(13,892, 14,124)$

To achieve statistical significance within each group, there was a minimum number of required samples, showed in the Column 2. In all cases, this number was respected, as reported in the Column 3. The number of bits used by the adversary to perform the attack successfully was summarized according to mean, standard deviation, and median. Considering that the mean is similar to the number of bits in p and the standard deviation is a small number, the data observed reveals that in all the attacks the number of bits required are within a restricted scope. The median coincides with $\log p$ in all cases. It shows a small variation in the number of bits demanded by the adversary to perform the attack, considering each group.

Confidence intervals, shown in the Column 7, were constructed with the data obtained in the experiment. It is possible to notice that in all intervals, the corresponding

[1]A complete description of the hypothesis tests and data summarization procedures used in this work can be seen in the book of Jain [Jain 1991].

value of bits in p is included. Thus, according to the zero mean tests performed, it implies in the absence of evidence to reject H_0 in all cases at the level of confidence of 95%. Furthermore, it can be noticed that the range of the intervals is narrow, not exceeding 1 bit. This indicates that the number of bits demanded by the adversary was estimated with a high degree of precision.

However, despite the accordance between the theoretic and practical results, it is not possible to generalized the results. More experiments should be carried because there are infinite possibilites of initialization of BM generators. The results observed just reinforce the predictions provided by the theoretical results.

The theoretical and experimental results discussed are also in accordance with Boyar [Boyar 1989] and Krawczyk [Krawczyk 1992]. According to these authors, under certain conditions, if the period of the generator is p, then it is possible to predict the generator's output with a number of guesses bounded by a polynomial in $\log p$.

3. The Generalized Quantum Permanent Compromise Attack

Quantum algorithms make use of parallelism, superposition, entanglement and other intrinsic characteristics to speed up the solution of certain problems when compared to the classical equivalent algorithms. This motivated the proposal of a permanent compromise attack to the Blum-Micali construction with Quantum Computing. The quantum algorithm presented in this section is an extension of the Guedes et al. algorithm to the BM generator [Guedes et al. 2010b].

The quantum generalized algorithm that performs the permanent compromise attack to the BM construction is composed by three registers, described as follows:

1. The space of the solution, containing $\lceil \log |\mathcal{D}| \rceil$ qubits initialized as $|0\rangle$, where $\mathcal{D}$ is the domain of the generator;
2. Ancillary qubits, one for each qubit in the first register, all of them initialized as $|0\rangle$;
3. Grover ancillary qubit, initialized as $|1\rangle$.

The proposed algorithm is composed of two main parts: the first part is responsible for the *identification of the representative* x_j of the internal state $X(j)$, as described in (7), and the second part performs the *amplitude amplification* of this element with the Grover's quantum search, increasing its probability to be measured. The quantum circuit showed in the Figure 1 implements the described algorithm, where its two parts are emphasized.

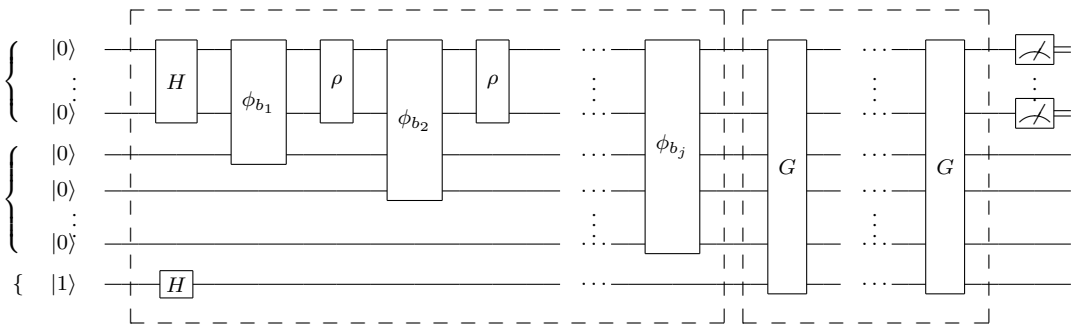

Figure 1. Quantum circuit that implements the generalized quantum permanent compromise attack to the Blum-Micali construction.

3.1. Identification of the Representative

The first part of the algorithm, composed by the gates H, ρ, and ϕ, characterizes a procedure to *mark* the representative $x_j \in X(j)$. The state that is the representative will be associated to $1 \ldots 1$ in the third register. To perform this procedure, firstly the Hadamard gate must be applied to the first and third registers, putting them in a equally distributed superposition.

The gates ρ and ϕ_{b_i} are defined in function of the one-way permutation and hard-core predicates of the generator under attack. They can be constructed as follows:

1. The ρ gate: Implements the one-way permutation of the generator. Since public parameters and the type of the generator are available, the adversary can rebuild the rule in (2). If the input is not in the domain $\mathcal{D}$, it must be left unchanged;
2. The ϕ_{b_i} gate: Is analog to the hard-core predicate for the generator. It determines which members of the first register in $\mathcal{D}$ could have generated the bit b_i. In the affirmative cases, it *flips* the last associated qubit in the second register. The actuation of the ϕ_{b_i} gate is similar to a conditional statement of the classical programming languages, but with the advantage of quantum parallelism.

After this first phase, the input state can be described as the following partition:

$$|\psi\rangle = \sin(\theta) \left|\psi_{x_j}\right\rangle + \cos(\theta) \left|\psi_{\mathcal{D}-x_j}\right\rangle \tag{8}$$

where $\left|\psi_{x_j}\right\rangle$ denotes the subspace of the representative (solution), and $\left|\psi_{\mathcal{D}-x_j}\right\rangle$ denotes the subspace of the elements there are not associated to $1 \ldots 1$ in the second register (non-solutions).

3.2. Amplitude Amplification

Since the representative is marked, an amplitude amplification must be performed to increase its probability to be measured. This amplification will be build with the Grover's quantum search algorithm – a procedure to perform search in unsorted databases. According to it, the amplitude of the solution (in this case, the representative) will be increased and the amplitude of the other elements will be decreased, due to the restriction of unitarity in quantum states. When a measurement is performed, the solution will be returned with high probability [Grover 1997].

The execution of k Grover's iterations can be summarized as:

$$G^k(|\psi\rangle) = \sin((2k+1)\cdot\theta) \left|\psi_{x_j}\right\rangle + \cos((2k+1)\cdot\theta) \left|\psi_{\mathcal{D}-x_j}\right\rangle \tag{9}$$

The required number of Grover's iterations will be discussed in the next subsection.

3.3. Required Number of Grover's Iterations

The required number of Grover's iterations k is derived from (i) the number M of marked states, i.e., states associated to $1 \ldots 1$ in the third register; and from (ii) the size N of the database, initialized as $N = 2^n$ where $n = \lceil \log |\mathcal{D}| \rceil$. This number k is given by:

$$k = \left\lfloor \frac{\pi}{4} \sqrt{\frac{N}{M}} \right\rfloor \tag{10}$$

where $\lfloor \cdot \rceil$ denotes the closest integer function. In the case of the BM generator, the number M can be obtained by approximation, using the result that there is only one marked state when the adversary knows $\log p$ bits from the generator, as showed in the Section 2.3.

3.4. Example

To illustrate the generalized permanent compromise attack to the BM construction, let's consider an attack to a BM generator. The adversary has knowledge of the public parameters $p = 7$ and $g = 3$, and has discovered three sequential bits $\mathbf{b} = 001$. The adversary will prepare the input of the algorithm in the following state, according to the rules of definition of each register:

$$|\psi_0\rangle = |000\rangle\,|000\rangle\,|1\rangle \tag{11}$$

The next step is the application of H gates in the first and third registers, resulting:

$$|\psi_1\rangle = \left[\frac{1}{\sqrt{8}}\left(|0\rangle + |1\rangle + \ldots + |7\rangle\right)\right]|000\rangle\,|-\rangle \tag{12}$$

After the identification of the representative phase, the state of the system will be the following:

$$\begin{aligned} |\psi_3\rangle &= \frac{1}{\sqrt{8}}\left(|0\rangle\,|000\rangle + |1\rangle\,|100\rangle + |2\rangle\,|110\rangle + |3\rangle\,|010\rangle + \right. \\ &+ \left. |4\rangle\,|001\rangle + |5\rangle\,|001\rangle + |6\rangle\,|111\rangle + |7\rangle\,|000\rangle\right)|-\rangle \end{aligned} \tag{13}$$

It can be noticed that the only state associated to 111 in the second register is the $|6\rangle$. Thus, the state $|\psi_3\rangle$ can be rewritten according to the following partition:

$$\begin{aligned} \left|\psi_3^{'}\right\rangle &= \frac{1}{\sqrt{8}}\,|6\rangle\,|111\rangle\,|-\rangle + \frac{1}{\sqrt{8}}\left(|0\rangle\,|000\rangle + |1\rangle\,|100\rangle + |2\rangle\,|110\rangle + \right. \\ &+ \left. |3\rangle\,|010\rangle + |4\rangle\,|001\rangle + |5\rangle\,|001\rangle + |7\rangle\,|000\rangle\right)|-\rangle \qquad (14) \\ &= \frac{1}{\sqrt{8}}\,|x_j\rangle\,|111\rangle\,|-\rangle + \sqrt{\frac{7}{8}}\,|\neg x_j\rangle\,|y\rangle\,|-\rangle \qquad (15) \\ &= \sin(\theta)\left|\psi_{x_j}\right\rangle + \cos(\theta)\left|\psi_{\mathcal{D}-x_j}\right\rangle \qquad (16) \end{aligned}$$

where $j = 3$; $|x_j\rangle = |6\rangle$; $\neg x_j$ denotes all states that aren't the representative x_j; $y \neq 111$; $\left|\psi_{x_j}\right\rangle = |6\rangle\,|111\rangle\,|-\rangle$; $\left|\psi_{\mathcal{D}-x_j}\right\rangle = |\neg\hat{x}_j\rangle\,|y\rangle\,|-\rangle$; and, $\theta \in \left(0, \frac{\pi}{2}\right)$ satisfies $\theta = \arcsin\left(\frac{1}{\sqrt{8}}\right) = 0.36$ radians.

The next step of the algorithm is the amplitude amplification. Before this step, it is necessary to determine how many Grover's iterations are necessary. The size of the space of the solution is $N = 8$, and $M = 1$ because just the state $|6\rangle$ is marked. Then $k = \left\lfloor \frac{\pi}{4}\sqrt{\frac{8}{1}} \right\rceil = 2$ iterations.

Therefore, 2 Grover's iterations on $\left|\psi_3^{'}\right\rangle$ will result:

$$\begin{aligned} |\psi_4\rangle &= G^2\left|\psi_3^{'}\right\rangle \qquad (17) \\ &= \sin((2\cdot 2+1)\cdot\theta)\left|\psi_{x_j}\right\rangle + \cos((2\cdot 2+1)\cdot\theta)\left|\psi_{\mathcal{D}-x_j}\right\rangle \qquad (18) \\ &= \sin(5\theta)\left|\psi_{x_j}\right\rangle + \cos(5\theta)\left|\psi_{\mathcal{D}-x_j}\right\rangle \qquad (19) \\ &= \sin(1.8)\left|\psi_{x_j}\right\rangle + \cos(1.8)\left|\psi_{\mathcal{D}-x_j}\right\rangle \qquad (20) \end{aligned}$$

A measurement in (20) will result in 6 with probability of $|\sin(1.8)|^2 \approx 94.83\%$. With this information the adversary successfully concluded the permanent compromise attack to the BM generator, endangering completely its unpredictability.

4. Related Work

Classical attacks to pseudorandom generators have been extensively explored by the literature. Kelsey et al. [Kelsey et al. 1998] proposed a taxonomy that classifies attacks to PRNGs in six different categories and also discuss their extensions. According to these authors, the study of cryptanalytic attacks on PRNGs has practical and theoretical interests because (i) there aren't any widespread understanding of the possible attacks to PRNGS; (ii) PRNGs are single point of failure in many real-world cryptosystems; and, (iii) many systems use badly-designed PRNGs or use them in ways to make various attacks easier than they need to be.

Regarding attacks to the BM construction, Sidorenko and Schoenmakers analyze the security of those generators against classical state recovery attacks, or permanent compromise attacks according to the Kelsey's taxonomy. They conclude that as the seed length increases, no polynomial-time adversary can retrieve the seed of the pseudorandom generator. Furthermore, they show that there are no tight reductions to this asymptotic statement [Sidorenko and Schoenmakers 2005].

Guedes et al. opened up the possibility to endanger the security of pseudorandom generators with Quantum Computing [Guedes et al. 2010b]. These authors proposed a quantum permanent compromise attack to the BM generator with a quadratic speedup over its classical counterpart. The present work generalizes this algorithm to the whole Blum-Micali construction and implies in the vulnerability of, at least, three different generators under quantum attacks – the BBS, Kaliski and BM pseudorandom generators.

Aside from the conclusions of Sidorenko and Schoenmakers, results to the BM generator showed that its security does not rely in the length of the seed, but in the public parameter p that define the domain $\mathcal{D}$.

5. Conclusion and Future Work

This paper introduced a generalized quantum permanent compromise attack to the Blum-Micali construction. This attack is based on Grover's quantum search and on quantum parallelism to recover the generator's internal state with high probability. Furthermore, this paper presented an experimental study that analyzed the number of bits an adversary needs to perform the attack successfully against the BM generator.

The proposed generalized quantum permanent compromise attack algorithm has complexity of $O(\sqrt{|\mathcal{D}|})$, contrasting with $O(|\mathcal{D}|)$ from the classical algorithm. This quadratic speedup over the classical equivalent solution is explained by the use of Grover's quantum search. Since the BM construction is widely adopted in real-world cryptosystems, the attack proposed may bring new security requirements to be imposed on pseudorandom numbers generators.

The reader is referred to [Guedes et al. 2010a] in order to obtain examples of the proposed attack against the BBS and Kalsiki generators.

In future works, in the case of the BM generator, the authors aim to analyze the adoption of some elements from the quantum algorithm to the discrete logarithm problem into the permanent compromise attack.

Acknowledgements

The authors gratefully acknowledge the financial support rendered by the Brazilian National Council for the Improvement of Higher Education (CAPES).

References

Blum, L., Blum, M., and Shub, M. (1986). A Simple Unpredictable Pseudo-Random Number Generator. *SIAM Journal on Computing*, 15:364–383.

Blum, M. and Micali, S. (1984). How to Generate Cryptographically Strong Sequences of Pseudorandom Bits. *SIAM J. Comput.*, 13 (4):850–864.

Boyar, J. (1989). Inferring Sequences Produced by Pseudo-Random Number Generators. *Journal of the Association for Computing Machinery*, 36:139–141.

Cloutier, D. (2005). Mapping the Discrete Logarithm. Senior thesis – Rose-Hulman Institute of Technology.

Gentle, J. E. (2003). *Random Number Generation and Monte Carlo Methods*. Springer.

Grover, L. K. (1997). Quantum Mechanics Helps in Searching for a Needle in a Haystack. *Physical Review Letter*, 79:325–328.

Guedes, E. B., de Assis, F. M., and Lula Jr., B. (2010a). Examples of the Generalized Quantum Permanent Compromise Attack to the Blum-Micali Construction. Available in http://sites.google.com/site/elloaguedes.

Guedes, E. B., de Assis, F. M., and Lula Jr., B. (2010b). Quantum Permanent Compromise Attack to Blum-Micali Pseudorandom Generator. In *Proceedings of the International Telecommunications Symposium 2010*.

Jain, R. (1991). *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modeling*. John Wiley.

Kaliski, B. S. (1988). *Elliptic Curves and Cryptography: A Pseudorandom Bit Generator and Other Tools*. PhD thesis, MIT, Cambridge, MA, USA.

Kelsey, J., Schneider, B., Wagner, D., and Hall, C. (1998). Cryptanalytic Attacks on Pseudorandom Number Generators. *Lecture Notes in Computer Science*, 1372/1998:168–188.

Krawczyk, H. (1992). How to Predict Congruential Generators. *Journal of Algorithms*, 13:527–545.

Marsaglia, G. (1995). The Marsaglia Random Number CDROM, including the DIEHARD Battery of Tests of Randomness.

Rukhin, A. (2008). A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Technical report, National Institute of Standards and Technology.

Shor, P. (1997). Polynomial-time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26:1484–1509.

Sidorenko, A. and Schoenmakers, B. (2005). State Recovery Attacks on Pseudorandom Generators. In *Western European Workshop on Research in Cryptology*, pages 53–63.