Algoritma Quantum Shor untuk Faktorisasi Bilangan Bulat

Andika Pratama (13507005)
Jurusan Teknik Informatika Institut Teknologi Bandung Jl. Ganesha 10, Bandung
email: if17005@students.if.itb.ac.id

Abstract *– Makalah ini mengulas bagaimana cara algoritma quantum shor menkomputasi faktorisasi bilangan bulat, hal yang sampai sekarang belum dapat dilakukan secara efisien oleh komputer konvensional, memnbuat faktorisasi bilangan bulat besar tidak feasible.*

Algoritma quantum shor dinilai penting karena secara teori, dapat digunakan untuk menfaktokan bilangan bulat dalam waktu polinomial. Dalam sejarah komputer quantum, algoritma ini telah banyak menarik perhatian dan mendorong perkembangan komputer quantum.

Kata Kunci*: algoritma shor, faktorisasi bilangan bulat, komputer quantum*

1. PENDAHULUAN

Menurut Teorema Fundamental Aritmatika, setiap bilangan bulat positif yang lebih besar satu memiliki faktorisasi prima yang unik[8]. Namun, teorema tersebut sama sekali tidak memberikan petunjuk bagaimana cara mencarinya, hanya bukti bahwa faktorisasi tersebut pasti ada.

Selama ini berbagai algoritma telah dikembangkan faktorisasi bilangan bulat. Namun sampai sekarang, faktorisasi bilangan bulat besar masih sebuah masalah yang sulit untuk dilakukan oleh komputer konvensional. Algoritma konvensional untuk faktorisasi bilangan bulat tercepat saat ini(*general number field sieve*) membutuhkan waktu sub-eksponensial:[1]

$$O\left(\exp\left(\left(\frac{64}{9}N\right)^{\frac{1}{3}}(\log N)^{\frac{2}{3}}\right)\right)$$

Kecepatan sub-eksponensial tersebut membuat waktu untuk mencari faktor dari bilangan bulat yang besar membutuhkan waktu yang begitu lama hingga dapat dianggap tidak *feasible.* Bahkan , walau tidak pernah dibuktikan, adalah anggapan umum bahwa untuk menemukan algoritma yang efisien(membutuhkan waktu polinomial) untuk faktorisasi bilangan bulat tidaklah mungkin.

Tidak adanya algoritma yang efisien membuat masalah ini dijadikan kekuatan kunci dari berbagai teknik kriptografi modern, dengan yang paling terkenal antara lain adalah teknik kriptografi Rivest-Shamir-Adleman(RSA).

Hingga pada tahun 1994, Peter Shor, seorang ilmuwan komputer menemukan algoritma quantum shor, yang dengan memanfaatkan sifat-sifat dari komputer quantum, sanggup menghitung faktorisasi bilangan bulat dalam waktu polinomial yaitu:[2]

$$O((\log N)^3)$$

Algoritma Shor yang memanfaatkan sifat komputasi quantum ini jauh lebih cepat dari algoritma konvensional yang ada, sehingga penemuan ini memicu perkembangan komputer quantum.

2. PEMBAHASAN

2.1 Komputer Quantum

Secara definisi, komputer quantum adalah komputer yang memanfaatkan fenomena-fenomena dari mekanika quantum, seperti *quantum superposition* dan *quantum entanglement* dalam proses komputasi data.

Komputer quantum dapat jauh lebih cepat dari komputer konvensional pada banyak masalah, salah satunya yaitu masalah yang memiliki sifat berikut:

1. Satu-satunya cara adalah menebak dan mengecek jawabannya berkali-kali
2. Terdapat n jumlah jawaban yang mungkin
3. Setiap kemungkinan jawaban membutuhkan waktu yang sama untuk mengeceknya
4. Tidak ada petunjuk jawaban mana yang kemungkinan benarnya lebih besar: memberi jawaban dengan asal tidak berbeda dengan mengeceknya dengan urutan tertentu.

Contoh dari masalah itu misalnya password cracker yang mencoba menebak password dari file terenkripsi(dengan asumsi passwordnya memiliki panjang maksimal).

Untuk masalah seperti diatas, waktu yang dibutuhkan oleh komputer quantum untuk menyelesaikannya proporsional dengan akar dari n. Hal ini dapat membuat waktu yang dibutuhkan dari tahunan menjadi hitungan menit.[3]

2.1.1 Qubit

Proses komputasi dilakukan pada partikel ukuran *nano* yang memiliki sifat mekanika quantum, maka satuan unit informasi pada Komputer Quantum disebut quantum bit, atau qubit. Berbeda dengan bit

biasa, nilai sebuah qubit bisa 0, 1, atau superposisi dari keduanya.

State dimana qubit diukur adalah sebagai vektor atau bilangan kompleks. Sesuai tradisi dengan quantum *states* lain, digunakan notasi bra-ket untuk merepresentasikannya.

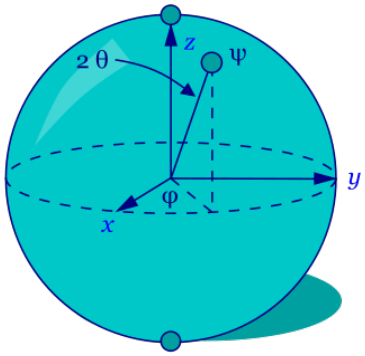
Pure qubit state adalah superposisi liner dari kedua state tersebut. Lebih jelasnya, sebuah *pure qubit state* dapat direpresentasikan oleh kombinasi linear dari *state*|0> dan *state* |1> :

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

Dengan α dan β adalah amplitudo probabilitas yan dapat berupa angka kompleks. α dan β dibatasi oleh persamaan:

$$|\alpha|^2 + |\beta|^2 = 1$$

State space dari sebuah qubit secara geometri dapat direpresentasikan Bloch sphere:

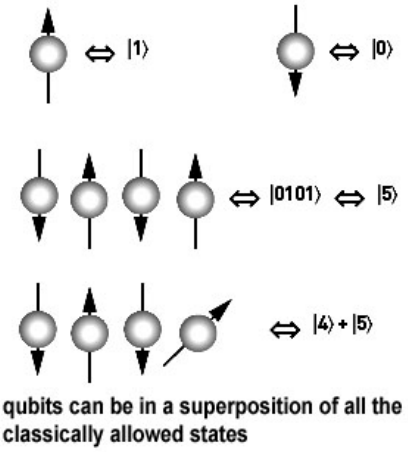


Gambar 1 : Representasi Qubit dengan Bloch Sphere

Ini adalah ruang 2 dimensi yang merupakan geometri untuk permukaan bola. Dibandingkan bit konvensional yang hanya dapat beradai di salah satu kutub, Qubit dapat berada dimana saja dalam permukaan bola.

Untuk penerapan fisiknya, semua sistem 2 level, selama ukurannya cukup kecil untuk hukum mekanika quantum berlaku. Berbagai jenis implementasi fisik telah dikemukakan, contohnya antara lain: polarisasi cahaya, spin elektron, muatan listrik, dll.

Contoh representasi qubit berdasarkan spin elektron: [4]



Gambar 2 : Repesentasi fisik Qubit dengan media spin elektron

Superposisi quantum adalah inti perbedaan antara qubit dengan bit biasa. Dalam keadaan superposisi, sebuah qubit akan bernilai |0> dan |1> pada saat bersamaan. Menurut interpretasi Copenhagen, bila dilakukan pengukuran terhadap qubit, maka hanya akan muncul satu state saja. State lainnya "kolaps" dalam arti hancur dan tidak mungkin diambil kembali. [5]

Pemanfaatan sifat superposisi qubit ini adalah Paralellisme Quantum. Paralelisme Quantum muncul dari kemampuan quantum register untuk menyimpan superposisi dari *base state*. Maka setiap operasi pada register berjalan pada semua kemungkinan dari superposisi secara simultan. Karena jumlah *state* yang mungkin adalah 2^n, dengn n adalah jumlah qubit pada quantum register, kita dapat melakukan pada komputer quantum satu kali operasi yang membutuh kan waktu eksponensial pada komputer konvensional.

Kelemahan dari metode ini adalah, semakin besar base state yang bersuperposisi, semakin kecil kemungkinan hasil pengukuran dari nilai hasil pengukuran tersebut benar. Kelemahan ini membuat pararellisme quantum tidak berguna bila operasi dilakukan pada nilai yang spesifik.

Namun kelemahan ini tidak begitu berpengaruh pada fungsi yang memperhitungkan nilai dari semua input, bukan hanya satu. Sebagaimana ditunjukkan pada Algoritma Shor.[6]

2.2 Dasar Teori Algoritma Shor

Algoritma Shor didasarkan dari sebuah teori bilangan: fungsi $F(a) = x^a \bmod n$ adalah feungsi periodik jika x adalah bilangan bulat yang relatif prima dengan n. Dalam Algoritma Shor, n akan menjadi bilangan bulat yang hendak difaktorkan..

Menghitung fungsi ini di komputer konvensional untuk jumlah yang eksponensial akan membutuhkan

waktu eksponensial pula. Pada masalah ini algoritma quantum shor memanfaatkan pararellisme quantum untuk melakukannya hanya dengan satu langkah.

Karena F(A) adalah fungsi periodik, maka fungsi ini memiliki sebuah periode r. Diketahui $x^0 \bmod n = 1$, maka $x^r \bmod n = 1$, begitu juga $x^{2r} \bmod n$ dan seterusnya.

Dengan informasi ini dan manipulasi persamaan sederhana berikut:

$$x^r \equiv 1 \bmod n$$

$$(x^{r/2})^2 \equiv 1 \bmod n$$

$$(x^{r/2})^2 - 1 \equiv 0 \bmod n$$

Dengan anggapan r adalah angka genap:

$$(x^{r/2} - 1)(x^{r/2} + 1) \equiv 0 \bmod n$$

Terlihat bahwa hasil dari $(x^{r/2} - 1)(x^{r/2} + 1)$ adalah kelipatan n. Maka selama $|x^{r/2}| \neq 1$, setidaknya salah satu dari $(x^{r/2} - 1)$ atau $(x^{r/2} + 1)$ memiliki faktor yang sama dengan n. Maka, dengan menghitung $\gcd(x^{r/2} - 1,n)$ dan $\gcd(x^{r/2} + 1,n)$ faktor dari n akan didapat.

Tetapi untuk menghitung r dari persamaan $x^r \equiv 1 \bmod n$ akan membutuhkan waktu eksponensial di komputer konvensional. Karena itu proses ini perlu dijalankan dengan komputer quantum agar seluruh nilai superposisi akan terhitung dalam sekali jalan.

Proses kunci lainnya adalah transformasi quantum fourier. Efeknya adalah akan meningkatkan puncak-puncak amplitudo probabilistik superposisi dimana probabilitas r adalah periode yang benar. Proses ini akan meningkatkan probabilitas kebenaran r.[6]

2.3 Langkah-langkah Algoritma Shor

Masalah yang hendak dipecahkan adalah: Diketahui sebuah bilangan komposit N, dicari sebuah bilangan bulat *x* dengan *x* bernilai $1 < x < N$.

.

1. Algoritma Shor untuk mencari faktor dari bilangan bulat n, dapat dipecah menjadi langkah-langkah berikut:
Menentukan apakan N adalah prima, genap, atau perpangkatan dari bilangan prima. Jika ya, maka Algoritma Shor tidak akan dipakai. Terdapat algoritma konvensional yang efisien untuk menentukan jenis n dan memfaktorkannya. Langkah ini akan dilakukan di komputer konvensional.

2. Ambil bilangan bulat q, dimana q adalah nilai dari perpangkatan 2 yang memenuhi: $n^2 \leq q < 2n^2$. Langkah ini akan dilakukan di komputer konvensional.

3. Mencari bilangan bulat random x yang relatif prima dengan n. Terdapat algoritma konvensional yang efisien untuk proses ini. Langkah ini akan dilakukan di komputer konvensional.

4. Membuat quantum register yang dipartisi menjadi dua bagian, katakan register 1 dan register 2. Register satu harus cukup besar untuk merepresentasikan bilangan bulat sebesar q-1, sedangkan register 2 sebesar n-1. Langkah ini perlu dilakukan di komputer quantum.

5. Masukkan kedalam register 1 superposisi yang setara dari semua bilangan bulat 0 sampai q-1, dan register 2 bilangan bulat 0. Langkah ini dapat dilaksanakan dengan menggunakan hadamard gates. Langkah ini perlu dilakukan di komputer quantum. Pada langkah ini, state dari quantum memory register adalah:

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a, 0\rangle$$

6. Aplikasikan fungsi $x^a \bmod n$ ke dalam semua bilangan bulat di dalam register 1, dan menyimpan hailnya di register 2. Karena prinsip pararellisme quantum, proses ini hanya perlu dilakukan sekali saja. Langkah ini perlu dilakukan di komputer quantum. Pada langkah ini, state dari quantum memory register adalah:

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a, x^a \bmod n\rangle$$

7. Mengukur dan memeriksa nilai yang disimpan pada register 2, mendapatkan nilai k. Langkah ini perlu dilakukan di komputer quantum. Langkah ini akan membuat register 1 semua superposisi state a dimana $x^a \bmod n \neq k$ "kolaps". Setelah langkah ini, state dari quantum memory register adalah:

$$\frac{1}{\sqrt{\| A \|}} \sum_{a' = a' \in A} |a', k\rangle$$

Dengan A adalah himpunan dari a' dimana $x^a \bmod n = k$, dan ||A|| adalah jumlah elemen dalam himpunan tersebut.

8. Menghitung transformasi quantum Fourier pada register 1. Langkah ini perlu dilakukan di komputer quantum. Sifat transformasi Fourier quantum akan mengubah |a> menjadi

$$|a\rangle = \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |c\rangle * e^{2\pi i ac/q}$$

Karena pararellisme quantum, langkah ini cukup dilakukan satu kali. State dari register setelah transformasi:

$$\frac{1}{\sqrt{\|A\|}} \sum_{c=0} \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |c,k\rangle * e^{2\pi i ac/q}$$

9. Ukur nilai state register satu, katakan nilai ini sebagai m. Bilangan bulat m memiliki probabilitas tinggi adalah kelipatan q/r, dimana r adalah periode yang kita cari. Langkah ini perlu dilakukan di komputer quantum.

10. Gunakan nilai m untuk mencari nilai r, dengan nilai m dan q diketahui. Jika r tidak dapat ditentukan, maka kembali ke langkah 4. Langkah ini akan dilakukan di komputer konvensional.

11. Setelah r diketahui, faktor n dapat ditentukan dengan menghitung gcd($x^{r/2}$ + 1, n) dan gcd($x^{r/2}$-1,n). Disaat ini faktor n telah ditemukan, dan Algoritma telah selesai. Langkah ini akan dilakukan di komputer konvensional.[6]

2.4 Kelemahan dan Kelebihan Algoritma Quantum Shor

Berbeda dengan komputer konvensional yang deterministik, komputer quantum bersifat non-deterministik dan probabilistik, yang berarti suatu algoritma kadang kala dapat berhasil dan kadang kala akan gagal biarpun untuk kondisi yang sama. Hal ini dikarena sifat pengukuran dalam mekanika quantum yang probabilistik. Akibatnya, Algoritma Shor dapat gagal menemukan faktor karena beberapa sebab, diantaranya:

- Hasil pengukuran dari transformasi quantum fourier dapat berupa 0, membuat langka ke 10 tak mungkin dilakukan.
- Kadang hasil faktorisasi algoritma akan menghasilkan 1 dan n, yang secara definisi benar tetapi tidak berguna
- Bila hasil r ganjil, maka langkah ke 10 tidak dapat dilakukan

Walaupun begitu, probabilitas sukses akan bertambah setiap kali algoritma diulang. Dalam Algoritma Shor yang dimodifikasi dengan penentuan order, probabilitas sukses setelah 2 kali jalan lebih dari 60%, dan probabilitas sukses setelah 4 kali jalan lebih dari 90%.[2]

Maka walaupun perlu berjalan untuk waktu yang tak ditentukan, waktu tersebut adalah polinomial. Lebih tepatnya, Algoritma Shor berjalan dengan kecepatan $O((\log n)^2 * \log\log n)$ pada komputer quantum, dan harus melakukan paska prosesing selama O(log n) pada komputer konvensional. Secara utuh algoritma ini polinomial.

3. HASIL SIMULASI DAN PEMBAHASAN

Simulasi Algoritma Quantum telah banyak dibuat untuk komputer konvensional, tetapi patut diingat bahwa simulasi-simulasi tersebut tidak akan bisa benar-benar meniru komputer quantum dengan sempurna, karena efek-efek eksklusif quantum seperti superposisi quantum dan quantum *entanglement*.

Program yang saya gunakan untuk simulasi dibawah adalah program shor yang dibuat olah Bernhard Oemer, ditulis dalam bahasa c++ dengan library QULIB untuk mensimulasikan komputer quantum abstrak.[7]

Disimulasikan proses Algoritma Shor dalam memfaktorkan bilangan bulat 15, karena 15 adalah bilangan bulat terkecil yang masih dapat difaktorkan oleh Algoritma Shor.

Hasil output program:

```
factoring 15: random seed = 7, tries = 1.
allocating 12 quBits with 256 terms.

RESET: reseting state to |0,0>
FFT: performing 1st Fourier transformation.
EXPN: trying x = 2. |a,0> --> |a,2^a mod 15>
MEASURE: 2nd register: |*,1>
FFT: performing 2nd Fourier transformation.
MEASURE: 1st register: |0,1>
<failed> measured zero in 1st register. trying again ...

RESET: reseting state to |0,0>
FFT: performing 1st Fourier transformation.
EXPN: trying x = 8. |a,0> --> |a,8^a mod 15>
MEASURE: 2nd register: |*,4>
FFT: performing 2nd Fourier transformation.
MEASURE: 1st register: |64,4>
rational approximation for 64/2^8 is 1/4, possible period: 4
8^2 mod 15 = 4. possible common factors of 15 with 5 and 3.
15 = 5 * 3.
program succeeded after 1 s and 2 iterations.
```

Pembahasan Simulasi:

Pada percobaan pertama, Algoritma Shor gagal karena hasil pengukuran 0 pada register pertama berarti q/r = 0 tidak memberikan informasi mengenai periode r.

Pada percobaan kedua, hasil pengukuran menghasilkan nilai dari puncak spektrum |64>. Dengan 64/256 = ¼ = q/r, periode r =4 diketahui dan kemungkinan faktor dari 15 telah ditemukan. Setelah faktor teronfirmasi, simulasi ini selesai.

Dari hasil simulasi tersebut terlihat sifat algoritma quantum yang non-deterministik, yaitu jumlah operasi yang harus dilakukan untuk berhasil tidaklah tetap. Namun secara probabilistas statistik, jumlah opersi yang dibutuhkan maksimal akan selesai dalam waktu polinomial.

4. KESIMPULAN

Karena kekuatannya yang dapat melakukan perhitungan eksponensial dalam sekali jalan. Komputer Quantum membuka banyak kemungkinan untuk memecahkan berbagai masalah yang sebelumnya dalam komputer konvensional dianggap secara praktis tidak mungkin. Kemampuan ini akan membawa dunia ke tahap perkembangan teknologi baru yang sedemikian tinggi. Karena itu, kegiatan pengembangan dan pengajaran komputer quantum perlu dicetuskan di indonesia.

Namun, begitu komputer quantum skala besar telah dibuat, maka dapat dikatakan bahwa teknik kriptografi yang bergantung pada faktorisasi bilangan bulat seperti RSA sudah ketinggalan jaman dan tidak aman lagi. Dengan begitu teknik kriptografi baru yang tidak mengantungkan kekuatannya pada kesulitan faktorisasi bilangan bulat perlu dikembangkan.

.

DAFTAR REFERENSI

[1]http://en.wikipedia.org/wiki/Integer_factorization ~diakses 1 Januari 2009

1[2]http://en.wikipedia.org/wiki/Shor's_algorithm ~diakses 27 Desember 2008

2[3]http://en.wikipedia.org/wiki/Quantum_computer ~diakses 27 Desember 2008

[4]http://en.wikipedia.org/wiki/Qubit ~diakses 27 Desember 2008

[5]http://en.wikipedia.org/wiki/Quantumsuperposition ~diakses 26 Desember 2008

3[6] http://alumni.imsa.edu/~matth/quant/299/paper/ 4~diakses 29 Desember 2008

5[7]http://tph.tuwien.ac.at/~oemer/doc/qcsim.pdf 6~diakses 3 Januari 2009

[8]Munir, Rinaldi, Diktat Kuliah IF2151 Matematika 7Diskrit, Departemen Teknik Informatika Institut 8Teknologi Bandung, 2004.

9Shor , Peter W, 1995, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*.