IB Computer Science Extended Essay

Will modern-day cryptographic measures and encryption methods be rendered useless by the computational power provided by emerging quantum computers?

Candidate Name: Patricio Lankenau
Candidate Number: 001510-027
Candidate Session: May 2013
IB Subject of Essay: Computer Science (HL)
Supervisor Name: Joel Mellen
Word Count: 3,824

Abstract

Research Question

As a computer science HL student, I learned about the importance of digital security. As we studied various methods of encryption I understood the importance of strong encryption in this modern age. Furthermore, when I read that the possibility of an entire new type of computer based on the rules dictated by quantum mechanics that could potentially be a threat to data integrity, I became interested in the possible implications that such an advancement of technology would bring. This gave rise to the research question, "Will modern-day cryptographic measures and encryption methods be rendered useless by the computational power provided by emerging quantum computers?"

Scope of Investigation

In order to approach this question, I performed a lot of research in current cryptographic systems. Using an array of sources including books in the Fondren Library at Rice and local publications, I developed an understanding of modern-day cryptography. Furthermore, I contacted my friend's father who works as a Computer Scientist for the Polish government under the sector of cryptography to gain insight into how cryptography is used in governments worldwide. After developing an understanding for current systems, I began my research into quantum mechanics and quantum algorithms. Since this field of studies is rapidly growing, most of the information came from peer-reviewed, published papers by doctors at the nation's leading universities such as Duke, MIT, and Princeton. In order to gain a better understanding of quantum algorithms, I took part in online lectures from MIT. As a way of comparing quantum algorithms with modern-day cracking algorithms, I wrote classical-computing methods and compared the theoretical runtimes.

Conclusion

From my research I found that the integrity of most modern-day cryptographic systems would be threatened to the verge of extinction by emerging quantum computers capable of performing previously-impossible tasks.

[Abstract Word Count: 296]

Table of Contents

Abstract 2

Research Question 2

Scope of Investigation 2

Conclusion 2

1. Introduction 4

1.1 Explanation of the Question 4

1.2 What is Cryptography 4

2. Modern-day Cryptography 4

2.1 Symmetric-key cryptography 4

2.2 One-way Hash Functions 5

2.3 Public-key Cryptography 5

3. Quantum Computers 6

3.1 Introduction to Quantum Computers 6

3.2 Potential of Quantum Computers 6

4. Quantum Cryptanalysis 7

4.1 Introduction to Quantum Cracking 7

4.2 Grover's Algorithm 7

4.3 Shor's Algorithm 8

5. Conclusion 10

5.1 Implications of Grover's Algorithm 10

5.2 Implications of Shor's Algorithm 11

5.3 Conclusion 12

5.4 Possible Solutions 13

Bibliography 14

1. Introduction

1.1 Explanation of the Question

The reason I chose this question, is the fact that its implications will be play a large role in the upcoming years. Our entire world is based on encryption. Whenever an email is sent, a file is saved, a database is accessed, a telephone call is made, even the way a car is opened, or an alarm is configured. We are in the age of information, and the integrity of our systems depends on cryptography to ensure security. With the emerging quantum technology operations which we considered impossible, such as breaking encryption methods, can be done in matters of seconds, essentially posing a threat to all information security and integrity.

1.2 What is Cryptography

Cryptography is an ancient art and science that has experienced many paradigm shifts, from simple letter substitutions to polyalphabetic substitutions, to rotor machines, to digital encryption to modern-day public-key cryptosystems. Cryptography is namely the study of techniques and applications which depend on the existence of difficult problems [RSA Labs]. Cryptography is crucial to our every-day lives, because it is the underlying foundation for the methods under which all digital information is communicated and stored. Using complicated computer algorithms, businesses are able to encrypt their sensitive data, and allow for secure transferring and storing of information. Using different methods of encryption and algorithms, readable plain-text (often referred to as cleartext) containing important and sensitive data can be encrypted or encoded into cyphertext, strings of characters which are unreadable. This secure string of characters can then be stored, or communicated without the risk of being read. Once received or when needed, based on the encryption algorithm, the cypher text can be decrypted (often using a security or private key) back into readable text, ready for use. The underlying security is based on the key generation, and hashing algorithms which make encryption un-reversible without the security key. To this day, there are many different hashing and encryption algorithms which are used by almost all methods of digital communication. Something as simple as rendering a page, or uploading a file, is most likely being transferred with some level of encryption. The principle for why modern-day cryptography works is because given the current day computational power, it is nearly impossible to reverse hashing and encryption algorithms, or decipher cyphertext in reasonable amounts of time.

2. Modern-day Cryptography

2.1 Symmetric-key cryptography

Symmetric-key encryption was one of the most popularly used methods to transfer digital information during the 20th century. Its wide-use was attributed to its simple implementation; however, it became apparent that emerging technology would render the encryption vulnerable and it is seldom used for sensitive information. Symmetric-key encryption is an algorithm which encrypts and decrypts information using a shared symmetric key, sometimes referred as a private key [Delfs, Hans]. The data is

encrypted using an algorithm based on the symmetric key, which generates unreadable cyphertext. When the data is retrieved the algorithm reverses the process, using the same symmetric key, to retrieve the original data.

This type of encryption is very fast, and albeit its insecure nature, is commonly used when large amounts of data are involved or when the data is to be decrypted shortly after encryption. Such an example are the Transport Layer Security (TLS) and Internet Protocol Security (IPSec) protocols which implement symmetric session keys with standard encryption algorithms to encrypt and decrypt confidential information as it is transferred between clients.

2.2 One-way Hash Functions

One-way hash functions are algorithms which are essentially impossible to reverse and simple to generate. This type of functionality is crucial to verifying integrity of files and storing information securely. A one-way hash function takes information, such as cleartext, and generates a unique hash text from it. The emphasis of these functions is that no two pieces of data should ever produce the same hash. Furthermore, a hash should not be able to be reverse-engineered to its cleartext form.

The two most widely used one-way hash functions are MD5 and SHA, developed in 1991 and 1995 respectively. The former is most commonly used to store user passwords, due to fast nature of encryption. The latter is currently the United States Federal Information Processing Standard, used by all communication within the US government [FIPS].

2.3 Public-key Cryptography

Public-key cryptography is based on public-key encryption, also known as asymmetric key encryption, which was invented by Whitfield Diffie and Martin Hellman in 1916 [RSA], as the replacement and solution to the insecure symmetric-key encryption methods. The biggest limitation of the symmetric-key algorithms is that because the sender and receiver have to possess the same private key, at some point in time the key has to be communicated. This required communication poses interception threats, and limits the encryption method.

Public-key encryption is based on two sets of keys, a private and a public one, per client. When a client sends information, the data is first encrypted using a symmetric key, and then the symmetric key is encrypted using the receiver's public key. The data can transferred securely because only the intended receiver is able to use his private key to decode the symmetric key required to decode the document. The principle for why this encryption method works is that the private and public keys are the product of large prime numbers, which are hard to generate quickly and therefore nearly impossible to crack.

This type of encryption is most commonly used to encrypt emails, transactions, and online payments and is the most used encryption algorithm in the world [RSA].

3. Quantum Computers

3.1 Introduction to Quantum Computers

The significance of computers has sky-rocketed in the past century; nearly three-fourths of households in the United States have at least one computer [ESA]. The modern-day computers are entirely based on the fundamental bit. This bit can represent two independent states, namely 0 or a 1; the former representing "off" and the latter "on". All data stored and transferred is fundamentally 0s and 1s, and all calculations including encryption and decryption, are limited by those two states and how quickly they can be accessed/changed.

Quantum computers fundamentally differ from classical computers in the sense that the fundamental building block, the quantum bit, or qubit can exist in the 0 state, the 1 state, or a coherent superposition of the two. This is feasible because of quantum mechanics which dictates that when a qubit is superimposed, it exists in two universes: in one as 0, and the other as a 1 [Bone]. Any calculation on such a superimposed qubit essentially acts on both values at the same time. This can be further enhanced by increasing the number of qubits. A calculation on a system with two superimposed qubits effectively performs the operation on all 4 states; a three qubit system performs the operation on 8 and a four qubit system on 16. By increasing the number of qubits, the quantum parallelism in the system exponentially grows. This parallelism, when used by quantum algorithms, can be used to perform tasks which would otherwise be impossible using classical computers.

3.2 Potential of Quantum Computers

Classical computers store information in series of bits, any operation is done through logic gates on the bits. Quantum computers manipulate qubits using quantum gates, similar to the way a classical computer would. The major difference, however, is that one operation on a bit affects only one state, whereas one operation on a superimposed qubit affects all of its states at once. Furthermore, when multiple quantum gates are applied successively, a quantum computer can perform complicated operation on all qubits in parallel [Polak].

The potential of parallel operation performed by quantum computers is that problems which would have been impossible using classical computers can be performed with ease using a qubit system. A quantum computer compromised of 500 qubits for example, can represent a quantum superposition of as many as 2^{500} states [CS Rice]. Each state is equivalent to a classical list of 500 bits representing 1's and 0's. Any calculation performed through quantum gates would therefore simultaneously operate on all 2^{500} states. This means that what operations that would require 2^{500} ticks (or rounds of operations) on a classical computer can be done in one tick using a quantum computer. Furthermore, when the quantum computer is probed for an answer, all 2^{500} states would collapse into a single quantum state corresponding to the result. Such a comparable operation would only be achievable with a classical supercomputer running 10^{150} individual processors running simultaneously [CS Rice].

4. Quantum Cryptanalysis

4.1 Introduction to Quantum Cracking

Cracking is most commonly referred as cryptanalysis which is the study of cryptographic systems, especially their hidden aspects and weaknesses. Cryptanalysis uses different methods to defeat encryption in order to gain access to the raw information, sometimes referred to as plaintext.

The methods used in cryptanalysis vary vastly depending on the information possessed and the technology available. The earliest forms of cryptanalysis were performed with pen and paper, but as new encryption methods were discovered, more sophisticated methods were required. Current-day cracking usually involves complex algorithms run on powerful supercomputers. Emerging computing power has made encryption safer, but has also allowed faster and more reliable methods for cracking.

Quantum Cracking refers to using the technology provided by quantum computers and algorithms to crack or decipher encrypted information. Although quantum cracking is mostly theoretical at its current stage, it poses a severe threat to modern-day cryptographic systems. The two most common encryption systems, AES and RSA, depend on the hardness of certain types of problems. But computational power provided by emerging quantum computers might just change the cryptographic systems as we know them.

4.2 Grover's Algorithm

One of the biggest challenges for computer scientist is searching for information. For example, if you have a database of names containing 1,000,000 entries that are sorted, finding the name you want would be arduous but fairly easy. However, if those 1,000,000 entries are random, the only way to find the desired name is to go through each entry individually [Bacon]. This means that the worst case scenario would require n steps but on average it would require $n/2$ steps. So for this database of 1,000,000 entries it would take 500,000 steps to find the desired name (on average).

This could all change with the introduction of Grover's Algorithm in 1996, designed and developed by Lov K. Grover. Grover's algorithm relies on quantum mechanical principles that would speed up the search from $n/2$ to $\sqrt{n}$ searches [Lavor]. This means that for our previous example, instead of 500,000 searches, the desired item could be found with just 1000 steps using Grover's algorithm.

Grover's algorithm is based on the fact that performing an operation on a qubit register will consequently perform that operation on all superpositioned states of the register. Whenever an operator known as the Grover diffusion operator is applied to the register, states which are not desired cancel each other out due to instability. After performing the operator a certain number of times the only state remaining will be the item that was searched for. This number of times is proven to be $\sqrt{n}$, which means that using this algorithm is nearly 100% more efficient than conventional factoring algorithms [Bacon].

Therefore the steps required to perform a Grover search can be summarized into three steps

1. Instantiate a quantum register in a superposition of all the possible states

2. Apply the Grover diffusion operator $\sqrt{n}$ times
3. Measure the resulting state

4.3 Shor's Algorithm

In 1994 Peter Shor developed an algorithm for factoring large numbers on a quantum computer. Factoring integers, especially large integers, is a very difficult task upon which most of modern-cryptography is based. The most efficient classical algorithm known to date has an exponential runtime. This means that the time required to factor an integer grows exponentially with the size of the input. This works as desired for small integers, but for large numbers is highly inefficient. Together with Daniel Simon, Peter Shor devised an algorithm that could factor large numbers in much smaller runtime than conventional algorithms.

The way that Shor's algorithm works is by reducing the factoring problem to finding the period of a function. Using quantum parellism, Shor's algorithm creates a superposition of all the values of the function in one step [Everitt]. Using mathematics beyond our scope, the amplitudes of the values are transformed and the period of the function is calculated. Once the period is obtained, it can be used to factor the original integer [Awwad].

The computer science explanation for how Shor's algorithm factorizes a number (in this example 15) is as follows.

The first step of the algorithm is to store all coherent superpositions of the number into a memory register. Coherent in this sense means that it essentially exists in two different universes. In the case of one qubit, it means that it exists as a '1' in one universe, and as a '0' in another. In order to store the number 15, we require four qubits. A four qubit register can thought of as existing in 16 universes at the same time. Each of the superpositions instances can be represented classically as follows:

0000	0110	1100
0001	0111	1101
0010	1000	1110
0011	1001	1111
0100	1010	
0101	1011	

Any calculation that is performed on the qubit register can be considered to be made on all of the different states simultaneously. Therefore even a simple calculation will be performed on every possible value that the register represents.

The second step of the algorithm requires a second qubit register (also with 4 qubits) and performs the following operation [Simon]:

$$Register\ 2 = X^{Register\ 1} \% N$$

In the operation, N denotes the number which we wish to factorize (in this case 15), X is a random discrete number smaller than N but larger than 1 that is chosen. The percent symbol signifies modulo which represent remainder division (the remainder of the division is the result).

The random integer, X, is raised to the power contained in the first register, divided by N, and then the resulting value is stored in the second register.

Like the previous step, the result will be a superposition of each of the different state's results.

The following table represents the results from performing the second step on 15 (N=15) using 2 as our random number (X=2)

First Register	Calculation	Second Register
0	$2^0\%15$	1
1	$2^1\%15$	2
2	$2^2\%15$	4
3	$2^3\%15$	8
4	$2^4\%15$	1
5	$2^5\%15$	2
6	$2^6\%15$	4
7	$2^7\%15$	8
8	$2^8\%15$	1
9	$2^9\%15$	2
10	$2^{10}\%15$	4
11	$2^{11}\%15$	8
12	$2^{12}\%15$	1
13	$2^{13}\%15$	2
14	$2^{14}\%15$	4
15	$2^{15}\%15$	8

As it can be noted, the resultant states in the second register are repetitions of the pattern 1, 2, 4, 8. This pattern is said to have a frequency of 4 since there are four values that are repeated.

This frequency is easy to calculate whenever the integers are small, such as in the previous example, but require a quantum computer for larger numbers.

The third step uses the previously calculated frequency to calculate a factor.

$$Factor = X^{f/2} - 1$$

In the previous example, the frequency was 4 so the calculated factor would be

$$Factor = 2^{4/2} - 1 = 2^2 - 1 = 4 - 1 = 3$$

And as we can confirm 15/3 results in no remainder therefore 3 is a factor of 15.

Although this is a simplified example, it is the fundamental reasoning behind why Shor's algorithm works. This method of factoring makes finding factors of large numbers (usually the result of two large primes) calculable in reasonable amounts of time compared to conventional methods which would require years to finish.

5. Conclusion

5.1 Implications of Grover's Algorithm

In order to compare the power of Grover's algorithm to classical algorithms for searches I wrote the following algorithm to search an unsorted String array for a specific value.

```
/*
 * Searches an array for a specific value
 * @precondition - Array exists and is unsorted
 * @return - Returns the index of the value, or -1 if not found
 */
public static int search( String[] database, String value){
	//create a for-loop starting at the first entry of the database
	for (int i = 0 ; i < database.length ; i++){
		//check if the entry at current index equals the desired value
		if (database[i].equals(value)){
			return i;
		}
	}
	//if the value was not found, return -1;
	return -1;
}
```

As mentioned earlier, this algorithm would require n steps in the worst case, and $n/2$ on average. This means that the classical search is reasonable for small database sizes, but not for large databases, which adds a layer of security for information stored in large unsorted databases.

That is, until Grover's algorithm was developed. With a time complexity (steps required) of $O(\sqrt{n})$ Grover's algorithm significantly speeds up searching for information [Barreno].

The following table compares various sizes of databases and the average number of steps required to find the desired item in each using the conventional method, and the quantum method.

N	**Grover Algorithm** $O(\sqrt{n})$	**Classical Algorithm** $O(n)$
10	3.16	10
1,000	31.6	1,000
1,000,000	1,000	1,000,000
1,000,000,000	31,600	1,000,000,000
1,000,000,000,000	1,000,000	1,000,000,000,000
1,000,000,000,000,000	31,600,000	1,000,000,000,000,000

It is clear that although the difference in steps is small when N is small (68% efficiency when $N = 10$), as N increases Grover's algorithm is highly efficient (97% efficiency when $N = 1000$ and 99-100% efficiency when $N \geq 10\text{^}4$).

The implications of this algorithm extend further than just finding information in databases. Because of the nature of Grover's algorithm, it can be applied to brute-forcing algorithms. A brute-force password cracking algorithm will try every single permutation (much like a classical database search algorithm) until it finds the correct one. Grover's algorithm can therefore be used to perform a quantum brute-force search in order to crack cryptographic systems such as AES or DES which use symmetric-key encryption.

Although Grover's algorithm only provides a quadratic speed-up compared to classical algorithms, if implemented by a quantum computer large and powerful enough, it could pose a severe threat to the US Government and many corporations world-wide that use AES as the standard for encryption.

5.2 Implications of Shor's Algorithm

As mentioned earlier, Public-key cryptographic systems are the most commonly-used methods of encryption around the world. They are easy to implement based on the assumption that large numbers (products of large primer numbers) are hard to factor even with the most advanced computer networks.

Shor's algorithm is capable of factoring large numbers in fractions of the time required using normal computing. Since factoring large numbers is what hinders code-breakers, Shor's algorithm could posses a catastrophic threat to the security and integrity of information around the world. As an example of the magnitude of the power of Shor's algorithm compared to modern-day computers I wrote an algorithm perform the prime factorization of numbers.

```
/*
 * Prints out all the factors of a number
 * @param a float
 */
public static void factor(float N){
    //creates a for-loop starting a 2 ending when the number is less than 1
    for (float i=2;N>1;){
        //if the number is divisible by the loop index
        if (N%i==0){
            //the loop index is factor
            out.print(i+" ");
            //divide the number by found factor
            N/=i;
        //if the number is not divisible by the loop index
        }else{
            //the loop index is not a factor, iterate to next step
            i++;
        }
    }
    out.println();
}
```

This algorithm's time complexity is $O(n)$ which means that the runtime is proportional to the number of digits being factored. This means that the algorithm is highly efficient with small to medium length numbers, but with large numbers (especially certain cases such as semi-primes or relative primes) the algorithm is really slow. For a number containing over 500 digits, the runtime would be in the orders of magnitude of years.

Shor's algorithm, on the other hand has a time complexity of $O((\log N)^3)$ which means that it is significantly faster for large numbers [Hayward].

A time comparison of Shor's algorithm versus a classical factorization algorithm is shown bellow.

N	**Shor's Algorithm** $O((\log N)^3)$	**Classical Algorithm** $O(n)$
10	1	10
1,000	27	1,000
1,000,000	216	1,000,000
1,000,000,000	729	1,000,000,000
1,000,000,000,000	1728	1,000,000,000,000
1,000,000,000,000,000	3375	1,000,000,000,000,000

As it becomes obvious, Shor's algorithm can factor extremely large numbers in very short amounts of time. In fact, it can factor a number with 1 billion digits faster than a classical algorithm could factor a number with a thousand.

This poses a large problem for public-key cryptographic system employed around the world. The world's most used and distributed cryptographic system is known as RSA encryption which relies entirely on the fact that the product of two large prime numbers is practically impossible to factor using classical computers. However, anybody with a quantum computer would be able to crack almost any RSA algorithm is seconds.

Although implementations of Shor's algorithm are currently mostly theoretical, in 2001 a team of researchers at IBM used a 7 qubit computer to perform the prime factorization of 15 [IBM].

5.3 Conclusion

Emerging computational power provided by quantum technology is changing the ways we think about encryption and computing. Although most of the advances have not been fully implemented yet, they pose a serious threat to the integrity of digital information. If a quantum computer with enough qubits is ever built, the two most common encryption algorithms AES and RSA would be vulnerable to cracking using Grover and Shor's quantum algorithms . The biggest drawback is the maintaining and stabilizing larger numbers of qubits in order to prevent decoherence which is the distortion of a quantum state caused me measurement or interaction [Polak]. Although this seems to be a physical limitation, some algorithmic attempts at quantum error correction have proven successful [Polak].

All things equal, if large enough quantum computers are ever built, modern-day cryptography will be rendered completely useless. This means that most information encrypting using the RSA encryption

algorithms can be cracked and accessed using Shor's algorithm. Furthermore, if an efficient quantum computer is ever created, the government standard for encryption, AES, could be compromised.

5.4 Possible Solutions

Although the focus of this paper is to evaluate the possibility of emerging quantum computers rendering modern cryptographic methods useless, it is important to note that there may be solutions. Considering that large-enough quantum computers are ever built and most modern-day cryptographic systems are rendered useless, it would not cause a cryptographic catastrophe. This paper focused on using the power provided by quantum computers to break current systems, but it must also be noted that there has been nearly as much development in the field of quantum cryptography. This field focuses on using quantum technology to develop secure cryptographic algorithms and systems [Sergienko]. There has been substantial progress in using polarized photons to transfer information securely; furthermore, using similar techniques to those of Shor's algorithm, quantum algorithms have been developed to encrypt information secure from quantum cracking [Gourley]. So although quantum computers will likely render modern-day cryptographic systems useless, they will also provide us with ways to encrypt and transfer information in new and more secure ways.

Bibliography

Barreno, Marco A.. "The Future of Cryptography Under Quantum Computers." *Dartmouth College Computer Science Technical Reports*. Dartmouth College, 21 July 2002. Web. 10 Apr. 2012. <www.cs.dartmouth.edu/~sws/theses/marco.pdf>.

Bone, Simon, and Matias Castro. "A Brief History of Quantum Computing." *Imperial College*. Imperial College London, n.d. Web. 6 Apr. 2012. <www.doc.ic.ac.uk/~nd/surprise_97/journal/vol4/spb3/>.

Bacon, Dave. "CSE 599d - Quantum Computing - Grover's Algorithm." Department of Computer Science & Engineering. University of Washington. University of Washington, Washington. 16 June 2006. Lecture.

Delfs, Hans, and Helmut Knebl. *Introduction to cryptography: principles and applications*. 2nd ed. Berlin: Springer, 2007. Print.

Everitt, Henry O.. *Experimental aspects of quantum computing*. New York: Springer Science+Business Media, 2005. Print.

Gourley, Bob. "Quantum Encryption vs Quantum Computing: Will the Defense or Offense Dominate?." *InfoSec Reading Room*. SANS Institute, n.d. Web. 3 Apr. 2012. <www.sans.org/reading_room/whitepapers/vpns/quantum-encryption-quantum-computing-defense-offense-dominate_720>.

Hayward, Matthew. "Quantum Computing and Shor's Algorithm." *Illinois Mathematics and Science Academy* 1 (2005): 1-61. Print.

"IBM News room - 2001-12-19 IBM's Test-Tube Quantum Computer Makes History - United States." *IBM - United States*. N.p., n.d. Web. 19 Nov. 2012. <http://www-03.ibm.com/press/us/en/pressrelease/965.wss>.

Lavor, Carlile, L.R.U. Massur, and Renato Portugal. "Grover's Algorithm: Quantum Database Search." *Cornell University Library* 1 (2003): n. pag. *Cornell University Library*. Web. 19 Nov. 2012.

Polak, Wolfang, and Eleanor Rieffel. "An Introduction to Quantum Computing for Non-Physicists." *ACM Computing Surveys* 1 (1998): 1-48. Print.

"RSA Laboratories - 1.2 What is cryptography?." *RSA - The Security Division of EMC*. RSA, n.d. Web. 22 Sept. 2012. <http://www.rsa.com/rsalabs/node.asp?id=2157>.

Sergienko, Alexander V.. *Quantum communications and cryptography*. Boca Raton, FL: Taylor & Francis, 2006. Print.

Awwad, Osama . "Shor's Algorithm." Explorations in Quantum Computing. Western Michigan University. Department of Computer Science, Michigan. 25 Sept. 2011. Class lecture.

"The FIPS Home Page." *Information Technology Laboratory Homepage*. United States Federal Government, n.d. Web. 22 Sept. 2012. <http://www.itl.nist.gov/fipspubs/>.

West, Jacob. "An introduction to Quantum Computing." *Department of Computer Science*. Rice University, n.d. Web. 23 Sept. 2012. <http://www.cs.rice.edu/~taha/teaching/05F/210/news/2005_09_16.htm>.