MIT COMPUTER USERS GROUP

NEWSLETTER

June 2017

Sharing knowledge
Bridging gaps

LETTER FROM THE EDITOR

Dear Readers,

We are delighted to bring to you the June ’17 issue of the MIT Computer Users Group Newsletter! We would like to begin by congratulating our newsletter team for their hard-work and dedication. The work wouldn’t have been so great without your inputs and co-operation!

We would like to express our gratitude towards **Prof U.K. Raut** for his guidance and support.

Here’s hoping this issue lives up to the MCUG tagline **“Sharing knowledge, Bridging gaps”**. Let’s come together and make this newsletter an iconic platform for revolutionary ideas and IT breakthroughs!

Thanks and Regards,
Ashwin Vaidya
Editor, MCUG Newsletter.

Editorial Team:

Pankaj Pandit
Gaurav Agarwal
Shaunak Joshi
Padmaja Pravin Saraf
Shweta Misra
Ankit Surkar
Bhargavi Dhagamwar

Follow us here:

https://www.facebook.com/mcugpune/
https://twitter.com/mitcug
https://www.youtube.com/user/MCUGPune

For any queries or feedback:
Mail us to: mcugnewsletter@gmail.com

Contact:
Ashwin Vaidya: 8826347839

INDEX

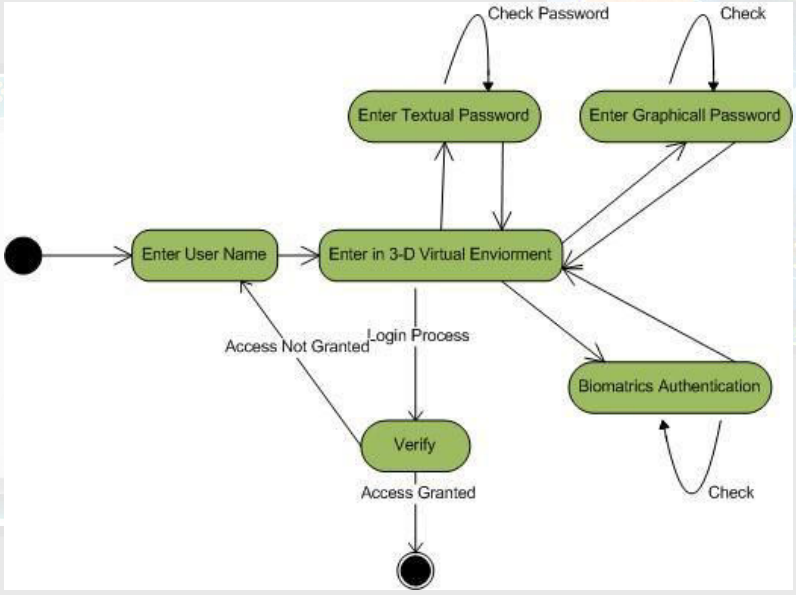
3D PASSWORD

Nowadays, security of a system is a major issue. Everything linked to a person from his/her identity to his/her bank account details is present over the internet. There is a constant fear of the possibility of a fraud.

WHY 3D PASSWORDS?

All the current authentication techniques are based on two authentication schemes, recall based and recognition based. Text based password recognition is an example of recall based scheme. You may have used it for your Gmail accounts. To secure system, the user provides a password to the system. Strong textual password can secure a system at a certain level. But it is difficult to memorize. In recognition base scheme, the user is required to identify and recognize his/her password which was created by him/her. You would be using your finger impression as your smartphone password. It is biometrics-based authentication technique. Biometrics authentication includes fingerprint, face recognition, voice recognition, etc. In this technique, various attacks, such as record attack are possible. Token-based authentication is also a part of recognition based scheme which includes ATM cards and smart cards. ATM cards can be stolen. In that case, there is a possibility of fraud and loss. Hence, considering security factor of the next level, 3D passwords are being proposed as the best solution.

WHAT IS 3D PASSWORD AUTHENTICATION?



3D password is a multi-factor authentication scheme. It combines all the existing authentication schemes such as recall based (textual passwords, etc.) and recognition based techniques (face recognition, finger impression, swipe cards, etc.) into a single 3D virtual environment. This 3D virtual environment consists of several objects with which the user can interact. The sequence of actions and interactions towards the objects inside the 3D environment constructs the user's 3D password. Interesting! Isn't it? It is the user's choice to select which type of authentication techniques will be the part of their 3D password. For example, if an object requests thumb impression and the user is not comfortable in providing such information, then the user may simply avoid interacting with that object. As this process gives user the freedom to choose which authentication techniques will be part of their 3D password, and a large number of items to interact with, the number of possible 3D passwords will increase. Hence, it becomes much more difficult for the attacker to guess the user's 3D password.

Consider a 3D virtual room having various items each of them having unique properties. The user will then interact with these properties accordingly. Each object in the 3D space can be moved around in an (x,y,z) plane. This property is common to all the objects in the

space. Suppose the user logs in and enters the 3D virtual room. He sees and picks up a newspaper initially at position (4,4,4) and moves it 4 places to right i.e. at (8,4,4). That can be identified as an authentication. Only the true user recognizes the object which he has to choose among many. This is the recall and recognition part of human memory coming into play. Also, a password can be set as approaching a radio and setting its frequency to a number that only the user knows. Security can be enhanced by the fact of including cards and biometric scanner as input. There can be levels of authentication a user can undergo**.** The more the confidentiality, the more complex it becomes. In that scenario, a virtual environment can be developed as a globe, a city or simply a room.

HOW ARE 3D PASSWORDS BETTER?

3D password authentication scheme is more useful & secure than any other authentication schemes. 3D passwords are considered stronger than other passwords. Following are some advantages:

1. It is a multi-factor authentication. There can be a combination of textual, graphical or recognition based password, etc. Even smart cards can be included.
2. As there can be almost unlimited number of passwords, hacking those are extremely difficult.
3. It gives the user the freedom to combine various authentication techniques.
4. It is hard to share these passwords.
5. They are easy to change.

WHERE CAN 3D PASSWORDS BE IMPLEMENTED?

3D passwords can be used in wide areas where more security is needed in the system. Some of the areas are as follows:

1. Networking: It provides better security to the server using this architecture. For email applications, 3D password is more secure and easier to use.
2. Nuclear & military areas: It is the most important area where high-end security is needed. We can use 3D password scheme in this area for providing more secure authentication and protecting data securely.
3. In addition, we can use 3D password authentication scheme in areas such as ATM, cyber cafes, industries (for data security), laptops or PCs, critical servers and web services.

-Bhargavi Dhagamwar
T.E. COMP-II

BLUE EYES TECHNOLOGY

'Perception' – perhaps the most distinctive character of all living being across the planet. What sets us apart from the insentient around us is our ability to sense and interpret our surroundings and inner selves. What if a machine had this capability? What if a computer could sense and interpret any situation and act accordingly? Doesn't machines being not only intelligent but also sensitive look like a farfetched thought? Well, going by the latest trends in the world of technology, not quite!

"Blue Eyes" was conceived by the research team of IBM at its Almaden Research Centre (ARC) in San Jose, California in 1997. Since then it has been a topic of great interest to computer enthusiasts. If it had to be explained in the simplest of words; Blue eyes technology makes a computer understand and sense human feelings and behaviour enabling it to react according to the sensed emotional levels. The aim of this is to give human abilities to a computer, so that it can naturally interact with human beings as with each other. All human beings have some perceptual capabilities, the ability to understand each other's emotional level or feelings from their facial expressions. Blue eyes technology aims at creating a computer that has similar capabilities of understanding the perceptual powers of human beings by recognizing their facial expressions and respond accordingly.

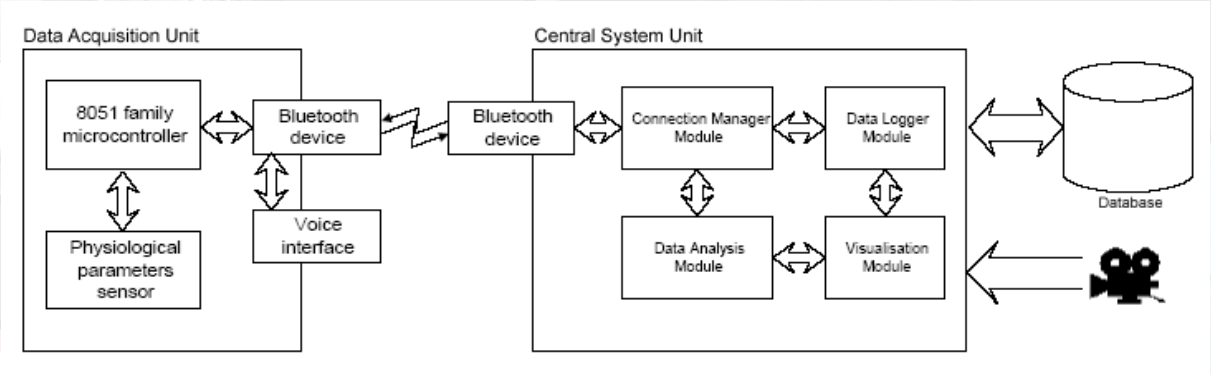
This technology majorly deals with facial recognition and speech recognition. It is a combination of hardware and software technologies. The major parts in a Blue eye system are Data Acquisition Unit and Central System Unit.

Data Acquisition Unit is a mobile part of the Blue eyes system. It's main task is to fetch the physiological data from the sensors and send it to the central system to be processed. To accomplish this task the device must manage a wireless Bluetooth connection (connection establishment, authentication and termination). Personal ID cards and PIN codes provide operator's authorization. Communication with the operator is carried out using a simple 5-key keyboard, a small LCD display and a beeper. When an exceptional situation is detected the device uses it to notify the operator. Voice data is transferred using a small headset interfaced with the DAU using standard mini-jack plugs. The various hardware modules that carry out the above mentioned functions include: System Core (Atmel 89C52 microcontroller), Bluetooth module (based on ROK101008), small LCD display (HD44780), EEPROM (24C16 - I2C, on a removable ID card), 13bit PCM codec (MC145483), Jazz Multisensor interface, Beeper and LED indicators, 6 AA batteries and Voltage level monitor.

Central System Unit hardware is the second peer of the wireless connection. It contains a Bluetooth module (based on ROK101008) and a PCM codec for voice data transmission. The module is interfaced to a PC using a parallel, serial or USB cable. The audio data is accessible through standard mini-jack sockets. To program the operator's personal ID cards a simple programming device has been developed. This device is interfaced to a PC using serial and PS/2 (power source) ports. Inside, there is Atmel 89C2051 microcontroller which handles UART transmission.

Blue Eyes software's main task is to look after the user's physiological condition. To assure instant reaction on any change in his condition, the software performs real time buffering of the incoming data, real-time physiological data analysis and alarm triggering. It comprises of several functional modules that allow flow of data from one element to another, like transfer of raw data from the Connection Manager to data analyzers, processed data from the data

analyzers to GUI controls, other data analyzers, data logger etc.



The system implies single-producer-multi-consumer thread safe queues. This basically means that the data being produced by the sensors of one user say the producer can be accessed by a number of other similar devices which can be referred to as the consumers. Any number of consumers can register to receive the data supplied by a producer. Every single consumer can register with any number of producers, receiving therefore different types of data. Every consumer may be a producer for other consumers. This approach enables high system scalability; new data processing modules (i.e. filters, data analyzers and loggers) can be easily added by simply registering as a costumer.

The Connection Manager is responsible for managing the wireless communication between the mobile Data Acquisition Unit the central system. Data Analysis module performs the analysis of the raw sensor data in order to obtain information about the operator's physiological condition. The module consists of a number of smaller analyzers extracting different types of information. The most important analyzers include saccade detector that monitors eye movements in order to determine the level of operator's visual attention, pulse rate analyzer which uses blood oxygenation signal to compute operator's pulse rate and the custom analyzers that recognize behaviours other than those which are built-in the system. The visualization module provides a user interface for the supervisors. It enables them to watch each of the user's physiological condition along with a preview of selected video source and related sound stream.

The information above is the jist of what seems to be yet another feather in the cap of technological advancement. Envision, a lovely world, where people team up with Computers! The PC can talk, listen or shriek so anyone might hear! We are indeed close to this world. The Blue Eye is indeed going to prove to be a golden chapter in the history of invention by mankind.

- Shweta Misra

B.E. COMP-II

BRAIN MACHINE INTERFACES

Remember the Matrix? Neo had these plugs connected to him which helped him enter the Matrix, where he could move, feel, fly or even stop bullets in mid air. Or the movie Avatar, where Jake lies in a capsule which helps him control the Avatar. Sure they are all science fiction, but are we really far off from having our brain directly connected to a computer? Well, yes and no. While research is being made toward restoring motor functions of the physically disabled by giving robotics limbs that they can control with their mind (so Avatar is covered), we are still far from entering our favourite game or any virtual environment (Matrix has to wait for a while).

This article attempts to explain how these technological feats are being achieved and where are we headed.

First, The Brain Machine Interface?

Think of brain like a remote control controlling your body. When you think of moving your arm, the brain sends a signal through the nerves passing through your spinal cord. The spine and the intermediate nerves can be thought analogous to a channel. Now imagine that you tap the connections in between and send and receive artificial signals. You could project a movie directly to your visual cortex without having to wear a VR headset. Or even better, send all motor functions to a computer and send all sensory functions from computer to brain and you will be able to physically (in a way as all the brain perceives of the physical world is from the senses) enter a game. This sounds sci-fi-i*sh,* but early uses of this technology have already shown results. This technology is known as Brain Machine Interface (BMI) or Brain Computer Interface (BCI).

BMI is the direct connection pathway between brain and an external device. It performs two functions: recording neurons and stimulating neurons. It is not always necessary to have bidirectional communication between the machine and the brain. Hearing aids for example only need to stimulate the neurons. There are three broad criteria in designing BMIs.

1. Scale: number of neurons that can be simultaneously recorded.
2. Resolution: how detailed is the information the tool receives—there are two types of resolution; spatial (how closely your recordings come to telling you how individual neurons are firing) and temporal (how well you can determine when the activity you record happened).
3. Invasiveness: is surgery needed, and if so, how extensively.

The Brain

To understand how to interface the brain with any device we have to know how the brain works. So, here is an oversimplified explanation of the brain. Brain is a densely packed complex structure. It is made up of hundreds of billions of neurons. Each neuron's axon has a negative "resting potential", which means that when it's at rest, its electrical charge is slightly negative. And it is connected to many dendrites of other neurons. These dendrites release neurotransmitters which raise or lower the charge on the neurons. If the neuron's charge rises over a certain threshold the neuron triggers. Now if we have a network of such neurons we get a neural network. We have many such networks in our brain which are responsible for various tasks. Let's call them modules for now.

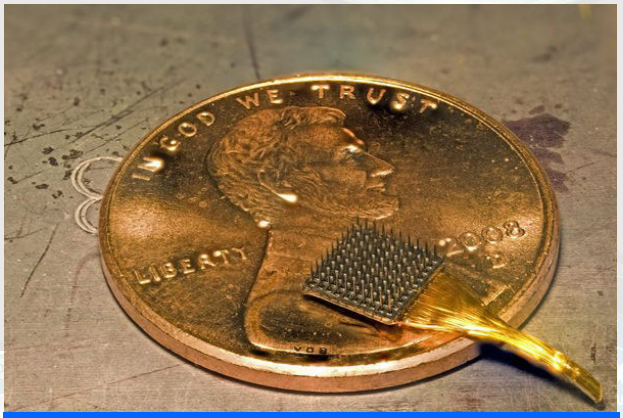
This is how Ray Kurzweil explains this in his 2014 TED Talk, " Consider a simple example.

We've got a bunch of modules that can recognize the crossbar to a capital A, and that's all they care about. They don't care if a song is playing or you are seeing a movie, but when they see a crossbar to a capital A, they get very excited and they say "crossbar," .That goes to the next level. Each is more abstract than the next one, so the next one might say "capital A." That goes up to a higher level that might say "Apple." Go up another five levels, and you're now at a pretty high level of this hierarchy, and stretch down into the different senses, and you may have a module that sees a certain fabric, hears a certain voice quality, smells a certain perfume, and will say, "My friend has entered the room." Go up another 10 levels, and now you're at a very high level. You'll have modules that say, "That was ironic. That's funny. She is pretty." You might think that those are more sophisticated, but actually what's more complicated is the hierarchy beneath them."

So now you would have an idea of how complicated the functioning of the brain is. The challenge lies in interfacing such a complex biological machine to a silicon based one. Well, researchers have found few means to interface the two and some of them are already in use.

How to read and stimulate neurons?

Consider the task of moving an artificial arm. You need to read the motor signals from the brain and send them to the computer which will tell the arm to move. This still does not solve the entire problem. The patient does not feel the artificial limb like it was his/her own arm, the way you would feel when say you lift your arm. This inability to be aware of the position of one's limb is called 'proprioception'. Overcoming this requires a feedback to the brain from the limb.



Micro-Electrode Array - brain implant

A commonly used but invasive (requires opening the skull) technique is to implant electrodes in the brain which can stimulate neurons, or record their activity. The non-invasive techniques can however only record.

A few examples of current generation BMI tools are, fMRI (Functional Magnetic Resonance Imaging), EEG (Electroencephalography), ECoG (Electrocorticography), LFP (Local Field Potential), Single-Unit Recording. A technology under research is 'neural lace'. It is an ultra-thin mesh that can be implanted in the skull, forming a collection of electrodes capable of monitoring brain function. To insert neural lace, a tiny needle containing the rolled up mesh is placed inside the skull and the mesh is injected. As the mesh leaves the needle it unravels, spanning the brain. Another idea is nano-ferrous dust that can be dissolved in water, which then reaches the brain through blood stream and on applying changing magnetic field heats up due to hysteresis. This then triggers its nearby neurons.

Possibilities

BMI technology is still in its infancy but, the future holds endless possibilities. We will able to restore the full motor functions of paraplegics or quadriplegics. Imagine having your brain connected to the internet. You will have access to all the information just by thinking. And to take it a little further, imagine being connected to an AI which helps you think when you need. You will be able to offload heavy thinking tasks to the AI without even realizing it. Even communication between individuals will improve as we will not be limited to using words and be able to send emotions directly. You will be able to store, share and re-experience your

experiences from the cloud. Live streaming will happen from person's point of view and you will be able to hear and smell whatever they hearing or smelling. And imagine having ability tweak internally, like removing addiction, depression and anxiety. Imagine representing something uncomfortable like pain to something like an auditory bell. Probably schools and colleges will not exist in the future. People will be expert in every field. You will be able to make yourself smarter by just downloading the required skills. You can then say like Neo, "I know Kung-Fu". And who knows, you might have subscription services like "On Demand Intelligence ".

- Ashwin Vaidya
T.E. COMP-I

GENETIC ALGORITHM

Traveling salesman problem! Yeah! We all are aware of it. To find a tour of given set of cities such that each city is visited only once, and simultaneously distance traveled is minimized. Nowadays problems like this are dealt very efficiently using the technique called as Genetic Algorithm (GA). We can see its real world implementation in the designing of neural network.

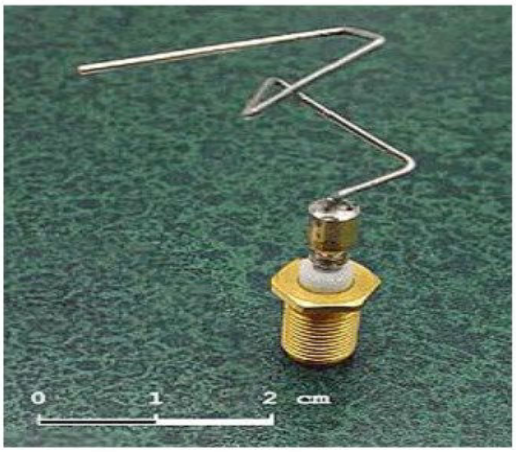
GA was invented by John Holland in mid 70's. It is a part of evolutionary computing and is a subfield of artificial intelligence. Evolutionary computing is a set of algorithms used for global optimization and is inspired by biological evolution. It is a technique used to solve problems for which optimization is required. It is based on Darwin's theory i.e. survival of the fittest. Nowadays, this technique is used in fields like stock market predictions, gaming, mathematics, etc.

There are various components in GA:

1. Search Space
2. Basic Algorithm
3. Encoding
4. Crossover & Mutation
5. Selection on the basis of fitness

How GA Works?

GAs are very easy to understand. Basic algorithm works until the optimal solution is reached. The first step in implementing GA is to create a population of N chromosomes or organism. Then we calculate fitness for each chromosome. For that, we have to develop a fitness function F(x). Fitness function calculates the fitness of each chromosome in the population. We then check whether the solution we get is optimal. If it is satisfactory then we stop the process otherwise we have to go for reproduction and kill weak ones. Reproduction is the generation of offspring from parents. After this, mutation is done. Mutation is a little change in DNA of offspring caused while copying genes from parents. While mutating, bit inversion or order changing of selected bit take place.



The 2006 NASA ST5 spacecraft antenna. This complicated shape was found by an evolutionary computer design program to create the best radiation pattern. It is known as an evolved antenna.

After mutation of the chromosome we again evaluate the fitness. If it is good enough, we stop the process, otherwise, we continue until we get the desired solution. With GA we look for best solution among number of possible solutions represented by a point in search space.

How Is Selection Done?

In GA while processing we have to select good (based on fitness) chromosomes out of a population of chromosomes. The selection is done on the basis of fitness of chromosomes. There are various methods by which selection is done.

In rank selection, we rank chromosomes according to their fitness. The worst chromosome is given rank as fitness 1, the second worst chromosome is ranked as fitness 2 and the fittest

chromosome is ranked as fitness N where n is the total number of chromosomes in population i.e. chromosome set. Another method of selection used in GA is roulette wheel selection.

Here, wheel is divided into different sections. The size of each section is proportional to the fitness value of chromosomes. In this method of selection marble is thrown on the wheel and where it stops, that chromosome is selected. Another Way of selection is to select some good chromosome with higher fitness and new offspring is created. Bad chromosomes (with lower fitness) are replaced new offspring. We keep doing this until we get an optimal solution. This process allows a large part of chromosomes to survive to next generation. This method is called as steady–state selection.

GA is an efficient technique areas which require optimal solution. In fields like strategy planning, sequence scheduling GAs are used on a wide level. By using this algorithm every time we get better and better answers, chances to get optimal solutions are more and hence they are very effective. Using GA's we can solve any optimization problem which can be described with chromosome encoding. It is helpful to solve problems with multiple solutions. GAs are easy to understand and practically doesn't demand the knowledge of mathematics.

With all these advantages GAs also have some disadvantages too. Its iterative nature differs it from rest of the machine learning algorithms. In real word problems like structural optimization, single fitness function may take several hours or days to complete simulation. Also, GAs do not deal well with complexity. i.e. in the problems where the number of chromosomes that are exposed to mutation are large, there is an increase in search space. This makes it difficult to use it on problems such as designing engine, building house or plane etc. To solve such problems with the evolutionary approach they must be divided into simplest representation such as encoding designs for blades instead of engines and building shapes instead of detailed construction plans. For some specific optimization problems, other machine learning algorithms may be more efficient than GA in terms of convergence speed.

In Spite of some disadvantages in this technique, GA can be used to optimize current and future health planning. For more effective use of GA lot of work is to be done. There is a lot of scope in this area like finding new crossover method, mutation method and finding effective fitness function. All of these efforts leads GA towards a very effective method for solving optimization problems in coming days!

- Padmaja Pravin Saraf
T.E. COMP-II

P=NP?

The **P versus NP** problem is a major unsolved problem in computer science. In fact, it is one of the seven “Millennium Prize Problems” selected by the Clay Mathematics Institute to carry a 1,000,000USD prize for the first correct solution. And before we go any further, I must stop you for the love of Alan Turing from putting the value of N as 1 and demanding a million dollars for it and proceed to escort you to the IT department where you belong.

So clearly P and NP are not variables. They are naming conventions to classify problems in computer science based on their difficulty measured in terms of time (number of steps) and space (amount of memory) required to solve the problem.

P- Polynomial time solving . Problems which can be solved in polynomial time, which take time like O(n), O(n2) and O(n3). So technically these problems require a lesser amount of resources of a computer due to their low complexity. For example: finding maximum element in an array, checking whether a string is palindrome or not, sorting a given list of numbers, getting a backlog in TOC for no reason and many more. You know, the easy stuff.

NP- Nondeterministic Polynomial time solving. Problem which can't be solved in polynomial time. These usually have exponential or other non polynomial complexities like O(2n) and O (n!). These become particularly more difficult to solve due to higher amount of resources needed to solve them. For example: Solving the Travelling Salesman Problem (you need to try almost all possible paths to find the shortest one) or solving the subset sum problem (you need to try all possible subsets within a given set to find the one whose sum is zero) or finding a girlfriend as a CS major (you most likely won’t find a solution). The difficult stuff.

However, NP problems are checkable in polynomial time which means that given a proposed solution of a problem, we can check that whether the solution is correct or not in polynomial time.

NP - Hard are a special category of NP problems which may not have solutions verifiable in polynomial time. They may not even be decidable i.e. they might not even have a solution. For example the subset sum problem as explained above - there may not be any subset which adds to zero.

To attack the **P = NP** question, the concept of **NP-completeness** is very useful. If a known NP -complete problem X can be reduced to problem Y in polynomial time and whose solution can still be verified in polynomial time, then Y is also said to be a NP-complete problem.

For example: we can reduce the Boolean SAT Problem (first NP-complete problem) to the Clique problem and prove the latter to be NP-complete as well.

The SAT problem: “Given a boolean expression, is it satisfiable?” which basically means to find a set of values that cause a given boolean expression to become true.

If the given expression 'x’ is “a&&(!b)”, then x is satisfiable (true) when a is true and b is false.

Figure 1 illustrates how the Clique problem (to find a subsets of vertices, all adjacent to each other, also called complete sub-graphs) can be reduced from the SAT problem:

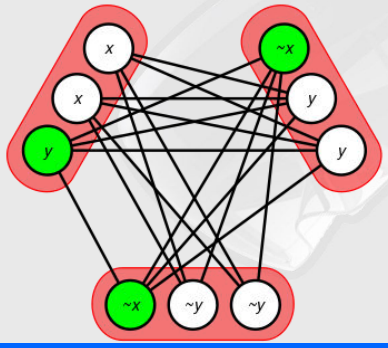


Figure 1 : The SAT instance (x&&x&&y)||(!x&&!y&&!y)||(~x&&y&&y) reduced to Clique.

Similarly bigger problems like the Travelling Salesman Problem can be proved as NP-complete as shown in figure 2.

Figure 2

If any NP-complete problem is in P, then it would follow that P = NP. Thus, if you solve any NP-complete problem in polynomial time, then you solve all other problems reducible to it in polynomial time (P=NP). Conversely, if you prove that there is no possible polynomial time solution to any of the NP-Complete problems, you prove that no polynomial solution exists for any of the NP-complete problems (P ≠ NP).

In either case you solve the great mystery of "P=NP?" and make yourself a million dollars richer and get crowned as the greatest computer scientist in all eternity.

Euler diagram for **P**, **NP**, **NP**-complete, and **NP**-hard set of problems. (Figure 3)

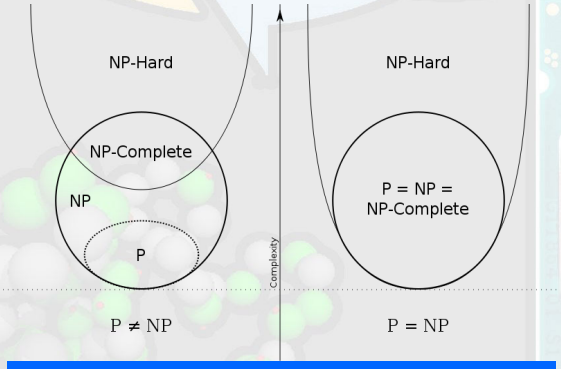


Figure 3

So, what happens when someone proves P ≠ NP? Well nothing much. People will probably just stop finding polynomial time solutions to all the NP-complete problems and go back to their sad lives.

However, if you prove P=NP, exciting things can happen. All the above mentioned problems could get solved in polynomial time meaning all known computer science problems could be solved immensely faster than ever before, public key cryptography would fail globally as there would be algorithms to get the keys in polynomial time and everything that depends on it would fail (including HTTPS and e-commerce infrastructure), Artificial intelligence would surpass human intelligence, the human genome could be decoded and we could eradicate all diseases known to humanity and make humans immune to any more to possible spring out in the future. The possibilities would be endless. Heck, even a CS major could finally find a girlfriend (Oh wait. That's still not happening. Refer NP-hard problems).

According to polls, most computer scientists believe that P ≠ NP. A key reason for this belief is that after decades of studying these problems no one has been able to find a polynomial-time algorithm for any of more than 3000 important known NP-complete problems. However no one has been able to give a valid proof for P ≠ NP either leaving scope for a proof for P=NP.

- Gaurav Agarwal

B.E. COMP-I

REAL SENSE

Have you all ever wondered while writing your computer programs “What the hell am I writing and is it really going to do?”, I guess this should be the basic question every budding programmer needs to ask himself to get the correct feel of programming and computing as it is imperative to know what is really happening when you use the computer.

Let us today try and find the answer to the above question and for that we need to know the brief history of computers and computations. In earlier days mathematicians used to do all the calculations manually and thus it always took a large amount of time to get a simple calculation done, but since the problems were not so enormous they dealt with it easily. But, there is an incident which is considered to be the turning point in mathematicians lives and that was during World War II, when British forces commissioned a team of mathematicians to decipher the messages sent by the German forces. It was a daunting task as the key used to code and decode the message changed every 12 hours and mathematicians were rarely able to recognize any messages. In this team there was the great man whom we know as Mr. Alan Turing who proposed that rather than doing such tedious work manually over and over again let a machine do it and recognize a pattern that can be used to decipher the text. Initially he was highly ridiculed for his vision as happens with all great minds but when this genius was able to develop a machine which we now know as the “Turing Machine”, a foundation for great future was laid.

The complete discussion tells us that Computer Science as we call it today is essentially a field of mathematics which was developed so as to reduce the complexity, time and human effort in mathematical computations. This fact has been completely neglected these days due to our curriculum which never told us that the algorithms and data structures we learn like Prim’s algorithm, binary search etc. that we use for our programs are basically the mathematical entities that a student or programmer should essentially know to build upon to solve a real problem.

Now if we come to programming there are some basic programming paradigms followed by the programmers like functional programming, imperative programming, logical programming etc. The programs that we traditionally write in C or C++ are the imperative programs where we are never concerned behind the mathematics related to our problem and that is what causes maximum problems in algorithm development as most of the programmers are never taught to look at the mathematical aspect of the matter.

But if we consider mathematics behind any basic computer problem the complete solution can be easily devised. And as from above discussion it is clear that Computer Science is a field of mathematics so all the computation problems can be represented as mathematical problems. Let us understand this by an example:

Program in C for fibonacci series-

```
int main() {
        int i, n, t1 = 0, t2 = 1,nextTerm;
        for (i = 1; i <= n; ++i) {
                printf("%d, ", t1);
                nextTerm = t1 + t2;
                t1 = t2;
                t2 = nextTerm;   }
        return 0;    }
```

Mathematically it can be represented as-

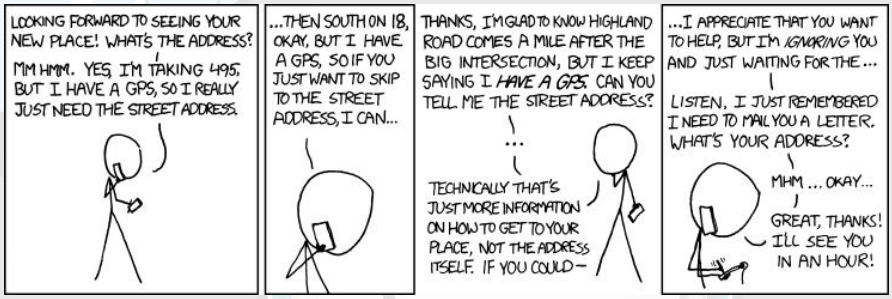
fib 0 = 0

fib 1 = 1

fib n = fib (n-1) + fib (n-2)

Where 'fib' is a mathematical function that collects all the values generated by the fibonacci sequence. From the above example it is evident that for writing a basic mathematical function, imperative programmers do a lot of headbanging whereas it can be easily solved as a mathematical function. All the assignments and data structures that are used in the C programs are the bindings that are provided to generate the result so that machines can understand the actual scenario and as machine only understands 0s and 1s, whatever you write needs to be converted to 0s and 1s and since that is the case why not to write the much easier mathematical functions and let the machine do all the job of assignment and binding as anyway the final aim of the programmer is to solve the problem and generate the output.

With this comes the concept of abstraction where the mathematical form is the near human form and the code written in form of 0s and 1s is a near machine form. It is highly evident that writing something in mathematical form is much easier and understandable and thus came up the concept of functional programming which is a programming paradigm which uses direct mathematical functions and representations. The compilers for these languages are so developed that all the bindings and machine implications are implicitly put in place and programmer needs to focus on the problem rather than the ways of implementation.



By definition in functional programming, programs are executed by evaluating expressions, in contrast with imperative programming where programs are composed of statements which change global state when executed. Functional programming typically avoids using mutable state. Functional programming has some intricate concepts like λ calculus, alpha functions, beta expansions, monads, Higher order functions, pure functions, immutable data, stateless computing etc. which need to be dealt with before going into depth but that is out of the scope of this article but we will discuss about it in next edition.

So, finally I would like to conclude with a hope that this article would put some light on the basic sense of programming, which is imperative to understand for any programmer. Mathematical vision of problem solving gives the programmer a broader aspect of problem and he is focused on the solution and not on the implementation as that will anyways follow when the solution is formulated. When programmer looks to find the solution, rather than going into the depth of resources needed and the bindings used in the implementation it becomes easy and viable for him to reach the final solution.

- Ankit Surkar

B.E. COMP-II

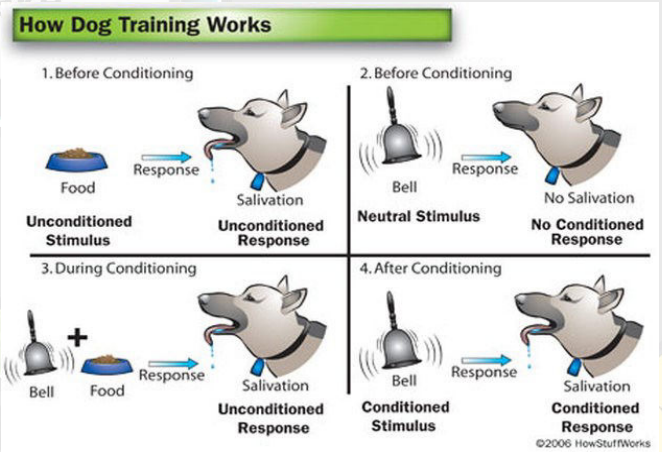
REINFORCEMENT LEARNING

We all know the AI buzzword that is the talk of the town nowadays not only in the field of computer science but also in variety of areas like astronomy, analysis of particle dynamics in particle accelerators, financial trading, healthcare, market personalization, recommendations of interested topics that you get in Google. Yeah it's 'machine learning'!

Simply put, machine learning is 'the ability of a machine or a program to learn based on the past data given to it doing so without explicitly programming the machine'. Amongst various machine learning techniques we have a very interesting technique called reinforcement learning.

Reinforcement Learning is similar to other ML techniques where the machine learns from the past dataset and further predicts a result on a particular input. The main difference is that in this technique a machine learns by an 'action' and a 'reward' system. Consider an example of a novice table tennis player. He/she knows a number of standard table tennis shots (backhand counter, forehand counter, top spin etc.) but doesn't know which one to play in response to an incoming ball. He knows a few standard shots and how they are played but doesn't know exactly which one to use at what time. Now he has to learn! To do this he maintains a score for every shot he hits and sets the initial value of each shot to zero. Now if he plays a shot, say TopSpin against a ball coming from the other side and if his hit is correct i.e. he could successfully land the ball on the other side, the score of that shot increases by whatever factor the player has decided. This is what we call as a 'reward' and the different shots that he knows are called 'actions'. If his hit is incorrect the score doesn't change. After he has played all the shots he analyzes the scores of each shot. The player keeps in mind the shot and the way of the incoming ball he received in which his score increased, so now he knows exactly when could he play that shot. So he has taken a feedback after playing all the shots at least once. In this manner the player plays a number of iterations finally learning the game. Here the player is called as the 'agent' or the machine and the type of incoming shot for the player is called an 'environmental state'. So finally, the score of an action is decided based on the correctness of output which depends solely on the combination of the environmental state and the action.

Consider environmental states as S and a set of actions as A. An individual action is given by a∈A . Here the goal of our agent is to maximize its total reward. As we saw with the player's example that reward is based on the function of the state and the action, this is represented by Q:S X A → R, where R is the reward.



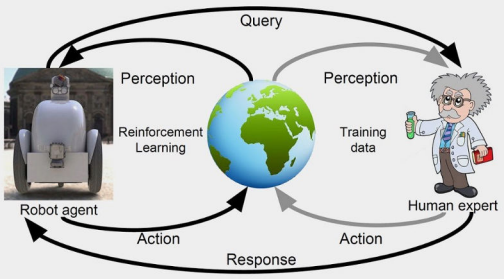
Training a dog by 'action' and 'reward' system

Q (Quantity function) is initialized to a random value before the learning starts. The agent then selects an action for a state. The agent then goes on to a new state such is determined as a function of the current state and the action taken, and depending on the new state the Quantity function is updated. This updation in every transition is meant to find the global optimum for Q. Hence each transition is defined as

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t . (r_{t+1} + \gamma . \max Q(s_{t+1}, a) - Q(s_t, a_t))$$

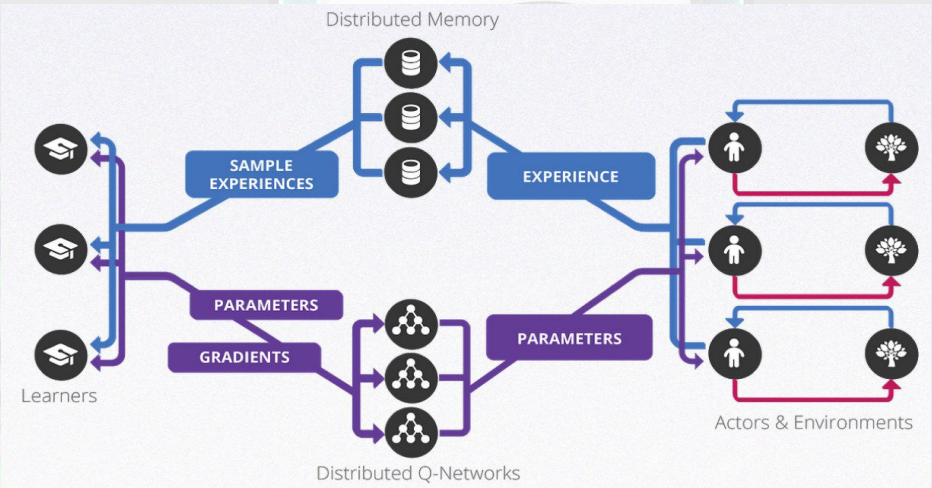
Q is represented by a concave graph (like parabola). Here r_t+1 is the reward observed after performing a_t in s_t ,and where α_t is called the learning rate ranging from 0 to 1. If α_t is very small then the algorithm has highly increased space and time complexity to find the global optimum resulting in a poor performance but if α_t is very high the amount of reduction of Q by every iteration is so high on a the concave graph of Q that after reaching a certain value closer to the global optimum the next iteration will lead to skipping of the global optimum resulting in never finding the optimum. Here γ is called the discount factor. The algorithm should be such that it not only optimizes Q for maximum current rewards but also consider about the future rewards a priori. So at each iteration generally the discount factor is increased progressively from a value slightly above than zero to the value slightly lesser than 1.

Well, this technique has stepped out of its theoretical bounds into the real world! A basic example is that of an industrial robot manufactured by a Japanese company called Fanuc. It implements an algorithm called Q-learning whose math we discussed above.



Basic Reinforcement Learning implementation in Robotics

Give the robot a task, like picking widgets out of one box and putting them into another container, and it will spend the night figuring out how to do it. Come morning, the machine should have mastered the job as well as if it had been programmed by an expert. It tries picking up objects while capturing video footage of the process. Each time it succeeds or fails, it remembers how the object looked, knowledge that is used to refine a deep learning model, or a large neural network, that controls its action.



Reinforcement Learning in a distributed system by DeepMind AI

Deep reinforcement learning agents have demonstrated remarkable progress on a wide variety of challenging tasks - from Atari to Labyrinth, from locomotion through manipulation, to poker and even the game of Go. Major industries like Google are striving to improve the capabilities of the RL agents, and to use them to make a positive impact on society.

- Shaunak Joshi
B.E. COMP-II


MC
UG

intel
Skylake