: N° d'ordre
: N° de série

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
Ministry of Higher Education and Scientific Research

UNIVERSITY OF ECHAHID HAMMA LAKHDAR - EL OUED

FACULTY OF EXACT SCIENCES
Computer Science Department

Thesis
submitted in partial fulfilment of the requirements for the degree of

ACADEMIC MASTER

Field: **Mathematics and Computer Science**
Option: **Computer Science**
Specialty: **Distributed Systems and Artificial Intelligence**

Presented by :

- **Laouid Brahim**
- **Tei Ahmed**

Title

An Efficient IoT Consensus Algorithm Based on Rounds competitions

Defended on June 20 th 2021

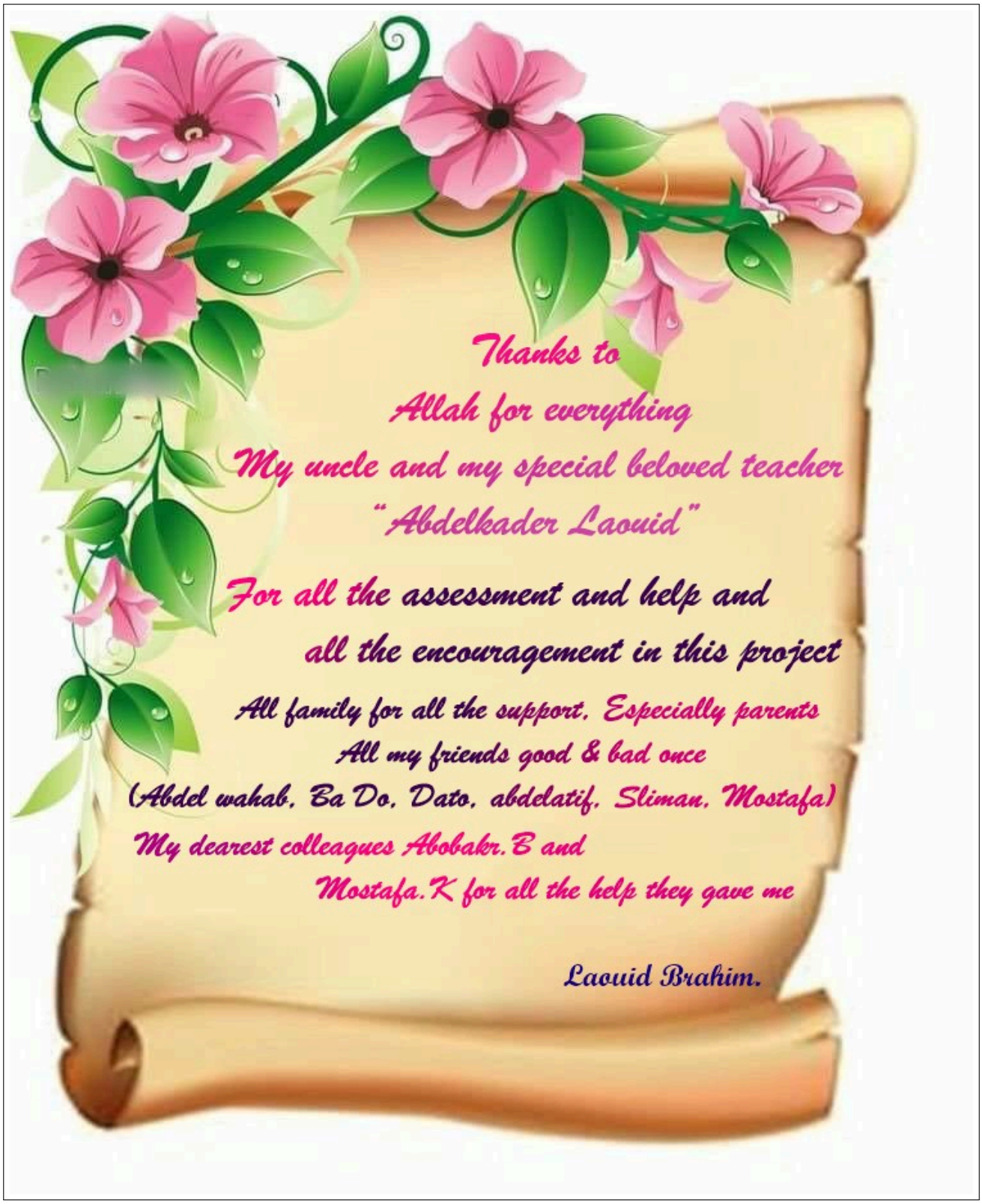
Examination Commitee :

M.	**Baggas Mounir**	MCA	Chair
M.	**Othmani Samir**	MAA	Examinator
M.	**Laouid Abdelkader**	MAA	Supervisor

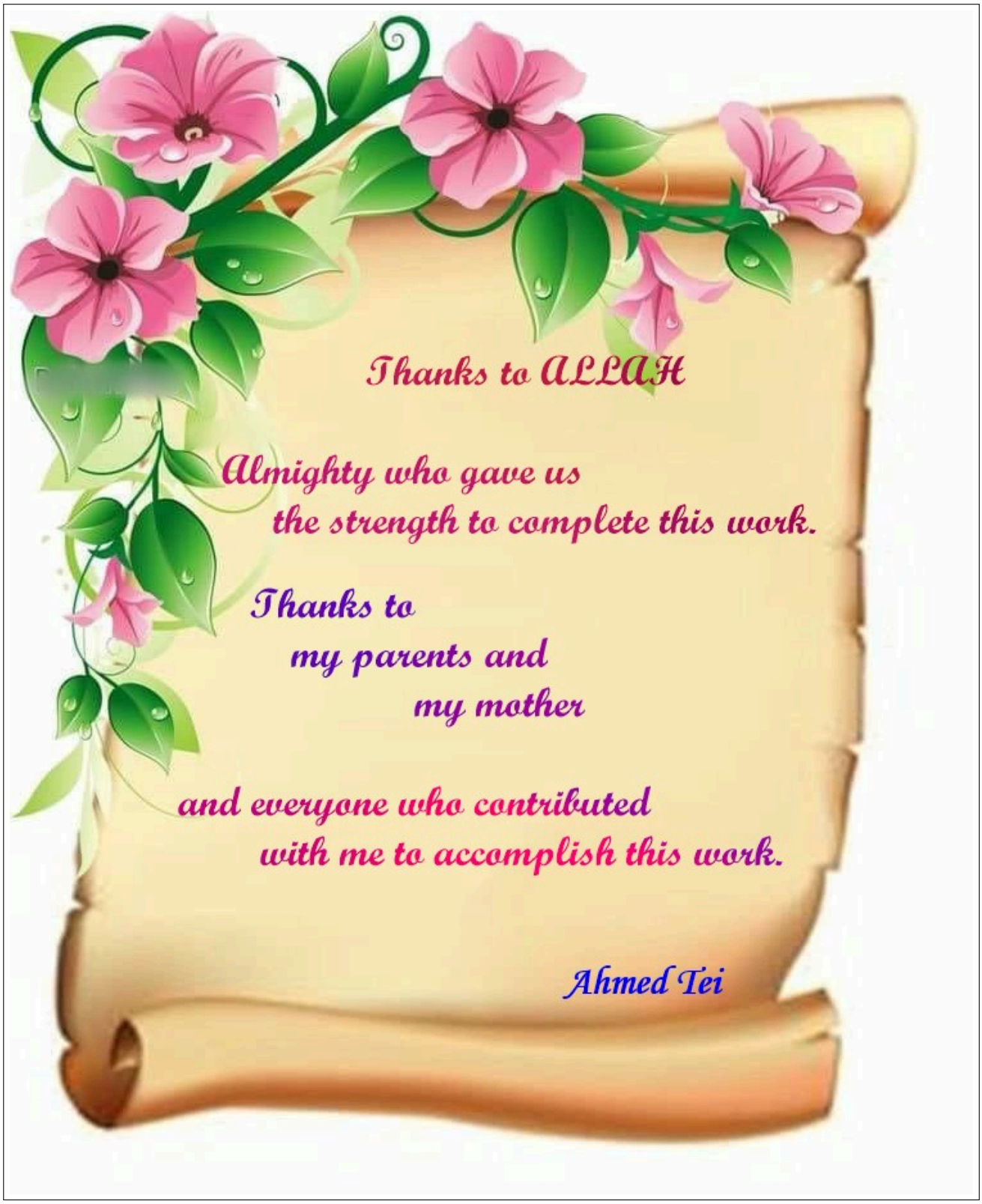
Academic year: 2020-2021

Acknowledgment

Dedicate


Thanks to
Allah for everything
My uncle and my special beloved teacher
"Abdelkader Laouid"
For all the assessment and help and
all the encouragement in this project
All family for all the support. Especially parents
All my friends good & bad once
(Abdel wahab. Ba Do. Dato. abdelatif. Sliman. Mostafa)
My dearest colleagues Abobakr.B and
Mostafa.K for all the help they gave me
Laouid Brahim.

Dedicate


Thanks to ALLAH
Almighty who gave us
the strength to complete this work.
Thanks to
my parents and
my mother
and everyone who contributed
with me to accomplish this work.
Ahmed Tei

Abstract

Several trusted tasks focus on consensus algorithms which are used to resolve agreement issues and to ensure the data reliability in untrusted P2P networks.

In many networking fields, the Proof of Work (PoW) consensus algorithm is frequently used. However, the pure PoW mechanism suffers from the weakness of the huge power consumption and prone against the vulnerability of 51 % attack. This work opts to mitigate the energy-inefficiency of the Blockchain Proof-of-Work (PoW) consensus algorithm by rationally repurposing the power spent during the mining process. The original PoW mining scheme is designed to consider one block at a time and assign a reward to the first place winner of a computation race. To reduce the mining-related energy consumption, we propose to compensate the computation effort of the runner(s)-up of a mining round, by granting them exclusivity of solving the upcoming block in the next round (he makes several rounds of proof). Any given node should resolve the game of the round N1 to participate in the round N. Furthermore, the nodes that will resolve the game of the round N cannot pass to the next round, i.e., N+1 , only when a predetermined number of solutions has resolved. This will considerably Profit waiting time with each round between different operations, and consequently a significant energy saving up to 50% could thus be achieved.

Key words : IoT, Consensus Algorithm, Blockchain, Peer to Peer, Energy consumption

Résumé

Plusieurs tâches de confiance exploitent des algorithmes de consensus pour résoudre les problèmes d'accord. Habituellement, les accords de consensus sont utilisés pour assurer l'intégrité et la fiabilité des données dans un environnement non fiable. Dans de nombreux domaines des réseaux distribués, l'algorithme de consensus Proof of Work (PoW) est principalement utilisé. Cependant, le mécanisme PoW pur a deux limitations principales, où la première est la faiblesse d'une énorme consommation d'énergie. La seconde limitation est la vulnérabilité de 51% d'attaque. Dans ce mémoire, nous cherchons à améliorer le protocole de consensus PoW en effectuant plusieurs séries de preuves. Tout nud de consensus donné devrait résoudre le jeu du tour en cours, c'est-à-dire $Round_i$, pour participer au tour suivant $Round_{i+1}$. Tout nud qui résout le jeu de $Round_i$ ne peut que passer au tour suivant, lorsqu'un nombre prédéterminé de solutions a été trouvée par d'autres nuds. Les résultats obtenus de cette technique montrent des améliorations significatives en termes de consommation d'énergie et font un état d'avancement du consensus. En fixant le nombre de processus, nous avons obtenu un taux de gain énergétique de 15,63% avec cinq tours et un taux de gain de 19,91% avec dix tours.

Mots cles : IoT, Algorithme de consensus, Blockchain, Peer to Peer, Consommation d'énergie

ملخص

تركز العديد من المهام الموثوقة على خوارزميات الإجماع التي تُســتخدم لحل مشــكلات الإتفاقية ولضــمان موثوقية البيانات في شبكات (p2p) الغير الموثوق بها.
في العديد من مجالات الشبكات ، يتم استخدام خوارزمية إجماع إثبات العمل(PoW) بشكل متكرر. ومع ذلك ، فإن آلية إثبات العمل النقية تعاني من ضعف استهلاك الطاقة الهائل وعرضة للهجوم بنسبة 51 ٪ . يختار هذا العمل التخفيف من عدم كفاءة الطاقة في خوارزمية الإجماع Blockchain Proof- of- Work (POW) من خلال إعادة تخصــيص الطاقة التي يتم إنفاقها أثناء عملية التعدين بشــكل عقلاني. تم تصــميم مخطط تعدين PoW الأصلي للنظر في كتلة واحدة في كل مرة وتخصيص مكافأة للفائز بالمركز الأول في سباق الحساب. لتقليل اســتهلاك الطاقة المتعلقة بالتعدين ، نقترح تعويض الجهد الحســابي للعدائين (المتســابقين) في جولة التعدين ، من خلال منحهم التفرد في حل الكتلة القادمة في الجولة التالية (يقوم بعدة جولات من الثبات). يجب أن تحل أي عقدة معينة لعبة الجولة N - 1 للمشاركة في الجولة N. علاوة على ذلك ، لا يمكن للعقد التي ستحل لعبة الجولة N أن تنتقل إلى الجولة التالية ، أي N + 1 ، فقط عندما تكون محددة مسبقًا عدد الحلول التي تم حلها. ســيؤدي هذا إلى ربح كبير من وقت االنتظار مع كل جولة بين العمليات المختلفة ، وبالتالي يمكن تحقيق توفير كبير في الطاقة يصل إلى 50٪.

الكلمات المفتاحية : إنترنت الأشياء ، خوارزمية الإجماع ، بلوك تشين ، نظير إلى نظير ، استهلاك الطاقة

CONTENTS

LIST OF FIGURES

GENERAL INTRODUCTION

In distributed systems there is a problem in its components on computers computers spread over different geographies and communicate by transmitting messages in order to achieve a common goal.

In this the work and with the absence of a global clock, the event of failure of an independent component in the system must tolerate the failure of individual computers. Each computer has only a limited and incomplete view of the system. Various hardware and software architectures are used for DS, For this we added a Peer-to-Peer (P2P) service where there are no special machines that provide services or manage network resources. Instead, all responsibilities are evenly distributed among all machines, called peers. Peers can serve as both clients and servers. In the context of DS.

In 1982 Appeared The Byzantine Generals Problem (BGP),That we shed light on in chapter 2 is a problem that was formalized by Leslie Lamport, Robert Shostak and Marshall Pease . The BGP is a metaphor that deals with the questioning of the reliability of transmissions and the integrity of the interlocutors. A set of elements working together must indeed manage possible failures between them. These failures will then consist of the presentation of erroneous or inconsistent information. The management of these faulty components is also called fault tolerance, which led us to talk about the synchronous and asynchronous system. Fischer, Lynch and Paterson (FLP) has shown that consensus cannot be reached in a synchronous environment if even a third of the processors are maliciously defective. In an asynchronous system, even with a single faulty process, it is not possible to guarantee that consensus will be reached (does not always end). FLP says consensus wont always be reached, but not that it ever will be. This study concerns asynchronous systems, where all processors operate at unpredictable speeds . In theory, consensus cannot be achieved systematically asynchronously. But, despite this result, it is possible in practice to obtain satisfactory results, for example, the non-perfect algorithms of Paxos Lamport (in the context of obvious failures and no Byzantine faults). Lamport, Shostak and Pease showed in , via the Byzantine generals problem, that: If f is the number of faulty processes, it takes at least 3f + 1 process (in total) for the consensus to be insured. Under these

conditions,

It guaranteed consensus algorithms, specifically Proof of Work, consensus perfectly, but its major problem is the enormous consumption of energy. In this work,we proposed a consensus protocol in an asynchronous environment. We control the energy consumption to ensure the synchronization bythe application of several rounds of consensus, in each round the nodes make a Proof-Wait, i.e., show the PoW then make await. And we organized this work as follows: In Chapter Two we talked about consensus algorithms, specifically we focused on Proof of Work, and in Chapter Four we present our proposed protocol,And how healthy it is to gain energy

CHAPTER 1

ENVIRONMENTS

1.1 An overview on peer-to-peer

1.1.1 Introduction

Peer-to-Peer networking is a hot buzzword (or as hot as it can be these days) that has been sweeping through the computing industry over the past years or so. Fortune crowned P2P as one of the four technologies that will shape the Internets future, and it continues to pop up in Some domains. Despite all the hype, there is a lot of substance to the topic as well [1].

1.1.2 What is P2P ?

The first keyword in the technical definition is peer-to-peer. This means that there is no central controller in the network, and all participants talk to each other directly [2].

The term peer to peer is understood in many different ways some of them are:

- Peer-to-Peer computing is a network-based computing model for applications where computers share resources via direct exchange between the participating computers.
- Peer-to-Peer (P2P) refers to a class of systems and/or applications that use distributed resources in a decentralized and autonomous manner to achieve a goal e.g. perform a computation.
- Wray defined Peer to peer networks just as a collection of heterogeneous distributed resources which are connected by a network.
- Peer-to-peer is a class of applications that take advantage of resources storage, cycles, content, human presence available at the edges of the Internet. Because accessing these decentralized resources means operating in an environment of unstable connectivity and unpredictable IP addresses, peer-to-peer nodes must operate outside the DNS and have significant or total autonomy of central servers. Thats what makes P2P distinctive [3].

1.1.3 Principles underlying P2P systems

The principle of sharing resources

All P2P systems involve an aspect of resource sharing, where resources can be physical resources : such as disk space or network bandwidth. logical resources : such as services or dierent forms of knowledge. By sharing of resources applications can be realized which could not be set up by a single node. This was the driving motivation behind a P2P system such as Napster.

The principle of decentralization

This is an immediate consequence of sharing of resources. Parts of the system or even the whole system are no longer operated centrally. Decentralization is in particular interesting in order to avoid

single-point-of failures or performance bottlenecks in the system. Examples of fully decentralized systems are Gnutella and Freenet.

The principle of self-organization

When a P2P system becomes fully decentralized then there exists no longer a node that can centrally coordinate its activities or a database to store global information about the system centrally. Therefore nodes have to self-organize themselves, based on whatever local information is available and interacting with locally reachable nodes (neighbors). The global behaviour then emerges as the result of all the local behaviours that occur.

1.1.4 How Peer-to-Peer (P2P) network works

In a standard centralized network when a user downloads a file, the user opens a web browser, visit the URL where the file is located and downloads the file. In this case, the website acts as a server and the user computer as a client that receives the data. This is the one-way data transfer where downloaded files are transferred from point A (the server) to point B (the user).

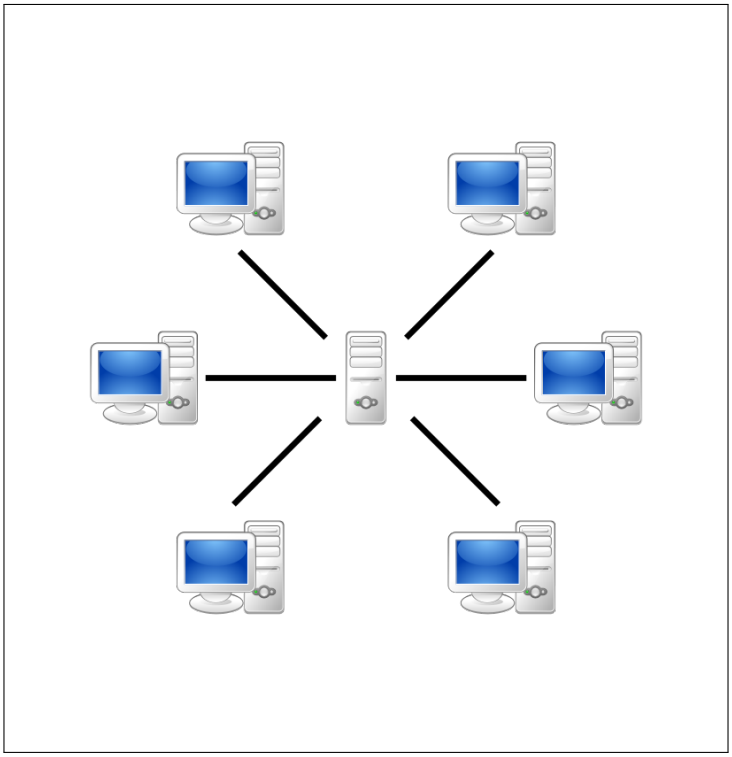


Figure 1.1: A network based on the client-server model.

When the same file is downloaded through a peer-to-peer network, the process is handled differently. Here the user has to pre-install peer-to-peer software, which creates a virtual network of peer-to-peer users when the user downloads the file is received in bits that are transferred from other peers in the network that already has the file. At the same time as the data is received, it is also sent

to the other peers in the network that ask for it. This situation is different because it is a two-way data transfer.

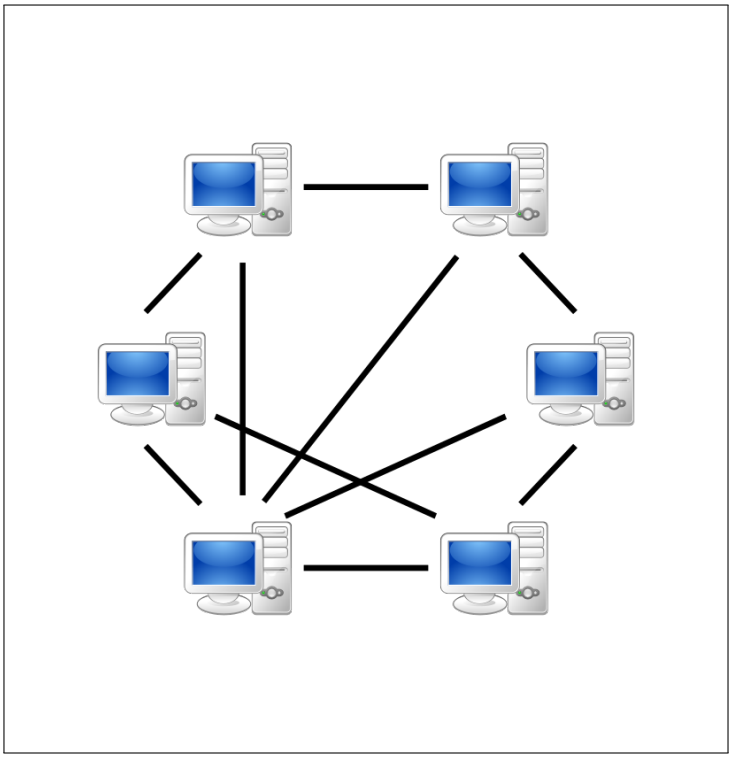


Figure 1.2: A Peer-to-Peer (P2P) network.

1.1.5 Advantages and disadvantages of pair to pair networks

Advantages

- No need for expensive server, and for network manager.
- Less network traffic than centralized client/server network.
- Extremly scalable, adding new peers is very easy.
- Difficult to take down,if you shot down couple of the peers, the other continue to work.

Disadvantages

- The data is not organized into specific shared area, sometimes it might be hard to locate the files.
- There is Little to no security [4].

1.1.6 Properties of pair to pair networks

- No centralized coordination.
- No centralized databases.
- No node has a global view of the system.

- All services and data are accessible from any node.
- The nodes are autonomous.
- Nodes and connections are unreliable.

1.1.7 Classification of P2P networks

It is also possible to group these architectures into 3 large groups often used to classify P2P networks:

Centralized P2P

There is a central server, which manages the identity of the users, the indexing of the shares, the searches and the linking of the peers: the server gives to each client the address of the peers possessing the searched file. The client then connects to this peer to exchange data directly. The files do not pass through the server. So the index is centralized, but the files are decentralized. This type of P2P is considered the weakest architecture because it is enough to attack the central server to bring down the network.

Advantages

- The ease of use is great, since there are no problems connecting to the right server.
- Document retrieval is facilitated. Indeed, the server is omniscient: if a file is available on the network, we are able to know it systematically.

Disadvantages

- Security is the Achilles' heel of this type of architecture: just delete the server and the entire network will be inactive.
- Sensitive to network partitioning (unreachable server) and attacks.
- No anonymity is possible, since each user is identified on the server. It is then very easy to develop user profiles.
- The risk of directory overload which can become a point of system failure.

P2P Hybrid

These networks use special nodes called "super-peer" that perform the indexing of shared files and serve as an intermediary for the nodes attached to them. They are chosen based on their computing power, their bandwidth

- Nodes with good bandwidth are organized in P2P. These are the super-Peers.
- Nodes with low bandwidth are connected in client/server mode to a Super Peer.

- Super-Peers have an index of the resources of their cluster. This type of architecture is particularly used in networks.

Disadvantage

- The risk of failure of a super-node.

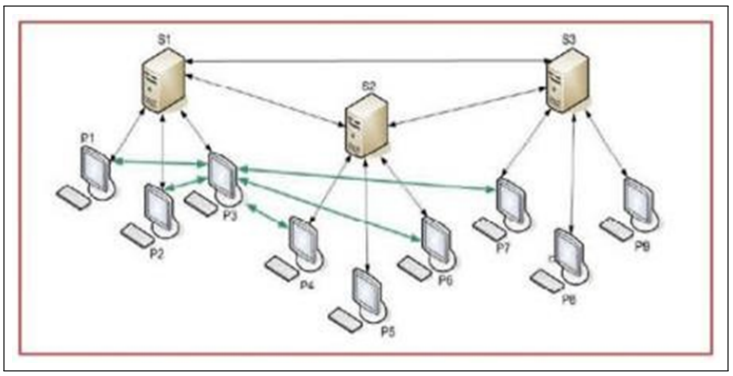


Figure 1.3: Example of a super-peer network.

P2P "pure"

Totally decentralized networks are called "pure". In this type of architecture, all nodes are equal and play the same role. Each peer thus manages resource searches and sharing, without going through a central server or super peer, generally using "flood" techniques or specific algorithms.

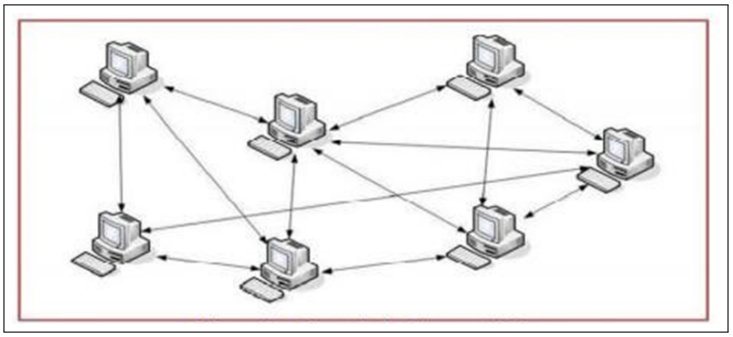


Figure 1.4: Example of pure P2P architecture.

Advantages

- The size of a network is theoretically infinite, unlike other architectures whose number of clients depends on the number and power of servers.
- The use of a network is anonymous, that is to say that it is impossible to locate someone there voluntarily.
- Very fault tolerant network.
- Adapts well to the dynamics of the network (paths and peers).

Disadvantages

- Large bandwidth consumer, high search time.
- No guarantee of success or estimate of the duration of requests.
- No security, no reputation (no notion of peer quality, no data provided).
- Problem with free riding (people not sharing files).

1.1.8 Characteristics of Peer-to-Peer architectures

P2P networks have three important characteristics :

- Clients are also servers and routers.
- Nodes contribute content, storage, memory, processor.
- Nodes are autonomous (no administrative authority)
- The network is dynamic: nodes enter and exit the network frequently.
- nodes collaborate directly with each other (not via well-known servers).
- Nodes have highly variable capabilities.

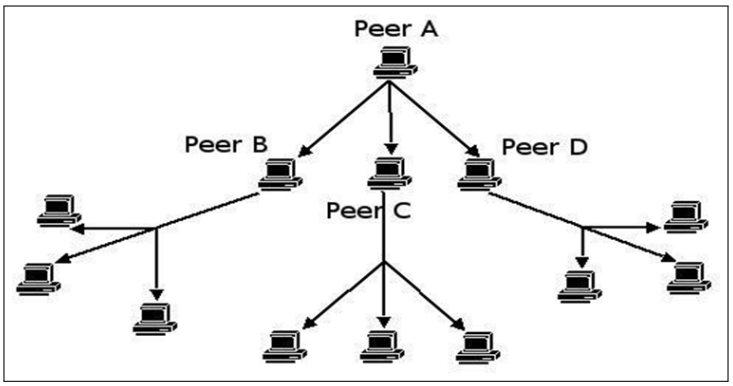


Figure 1.5: Peer-to-Peer architectures

1.2 Distributed system

1.2.1 What is a distributed system?

A distributed system is a software system in which components located on networked computers communicate and coordinate their actions by passing messages, that appears to its users as a single coherent system The components. interact with each other in order to achieve a common goal. This definition refers to two characteristic features of distributed systems.

The first one : is that a distributed system is a collection of computing elements each being able to behave independently of each other. A computing element, Which is indicated generally refer to as a node, can be either a hardware device or a software process.

second element : is that users (be they people or applications) believe they are dealing with a single system. This means that one way or another the autonomous nodes need to collaborate. How to establish this collaboration lies at the heart of developing distributed systems [5].

1.2.2 Distributed system theory

The first is the impossibility theory of FLP, that is, under the premise of a reliable network, it is impossible to have a deterministic algorithm to solve the consistency problem in any node failure, or the consistency problem of one or more minimized asynchronous model systems.

The second impossible principle is the CAP principle. Especially in traditional distributed systems, we are all very familiar with this theory. For example, in practice, it is impossible to guarantee strong consistency, availability and partition tolerance in a system at the same time, but to weaken one feature to ensure the other two based on engineering characteristics.

The third is the BASE theory. It is an extension of CAP principle, like CAP theory with strong consistency; BA stands for basic usability, S for a soft state, and E for final consistency [6].

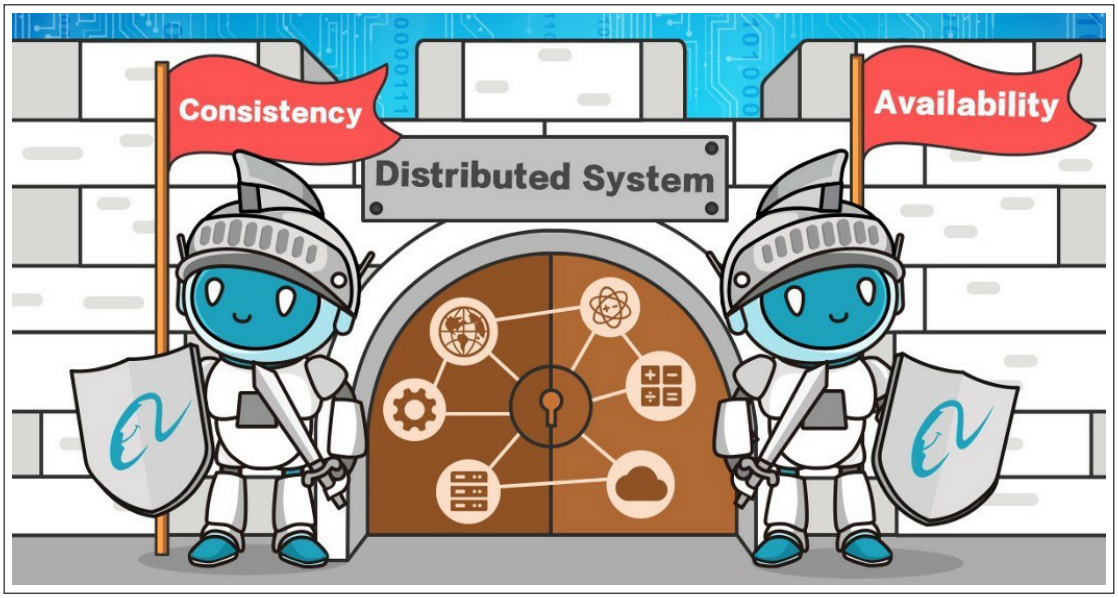


Figure 1.6: Distributed system theory.

1.2.3 Centralized system vs Distributed system

Criteria	Centralized system	Distributed System
Economics	Low	High
Availability	Low	High
Complexity	Low	High
Consistency	Simple	High
Scalability	Poor	Good
Technology	Homogeneous	Heterogeneous
Security	High	Low

Figure 1.7: Centralized system VS Distributed system.

1.2.4 Types of distributed systems

Distributed computing systems

- Cluster computing systems
- Grid computing systems
- Cloud computing

Distributed information systems

- Transaction processing systems
- Enterprise application integration

Distributed pervasive systems

- Ubiquitous computing systems
- Mobile computing systems
- Sensor networks [7].

1.2.5 Characteristics of a distributed system

A distributed system must possess the following characteristics to deliver utmost performance for the users :

Fault-tolerant

Due to a distributed system having many computers comprised of different aged hardware, it is very likely for a part to fail in such a way that a node can no longer operate. **Fault tolerance** is the ability for the system to handle such failures, this is achieved by using recovery and redundancy. Recovery is where a component will act in a predictable, controlled way if it relies on a component. Redundancy is where crucial systems and processes will have a backup that takes over if a system fails.

Scalability

Scalability is one of the major characteristics that effectiveness of a distributed system, it refers to how easily the system can adapt to a changing size. This is due to the volatile nature of computers, such that a device is prone to leaving and joining the system at will. This volatility is caused by computers powering down, or unstable networks causing connectivity issues.

The more nodes that try to communicate or use this component, the more likely it is that there will be a bottleneck at this point in the system.

Openness

The openness of a distributed system is defined as the difficulty involved to extend or improve an existing system. This characteristic allows us to reuse a distributed system for multiple functions or to process varying sets of data.

Security

Distributed systems should allow communication between programs users resources on different computers by enforcing necessary security arrangements. The security features are mainly intended to provide confidentiality, integrity and availability. **Confidentiality** (privacy) is protection against disclosure to unauthorised person. **Integrity** provides protection against alteration and corruption. **Availability** keeps the resource accessible.

Heterogeneity

Heterogeneity refers to the ability for the system to operate on a variety of different hardware and software components. This is achieved through the implementation of middle-ware in the software layer. The goal of the middle-ware is to abstract and interpret the programming procedural calls such that the distributed processing can be achieved on a variety of differing nodes

Concurrency

Concurrency refers to the systems ability to handle the access and use of shared recourses. This is important because if there is no measure implemented it is possible for data to get corrupted or lost by two nodes making different changes to the same resource such that the system can carry this error through different processes causing an incorrect result.

One way to counteract these errors is to implement a locking mechanism making a node unable to access a resource whilst it is being used by another node.

Transparency

Transparency in a distributed system refers to the idea that the user perceives that they are interacting with a whole quantity rather than a collection of cooperating components. Transparency can be split into eight sub-characteristics defined like following:

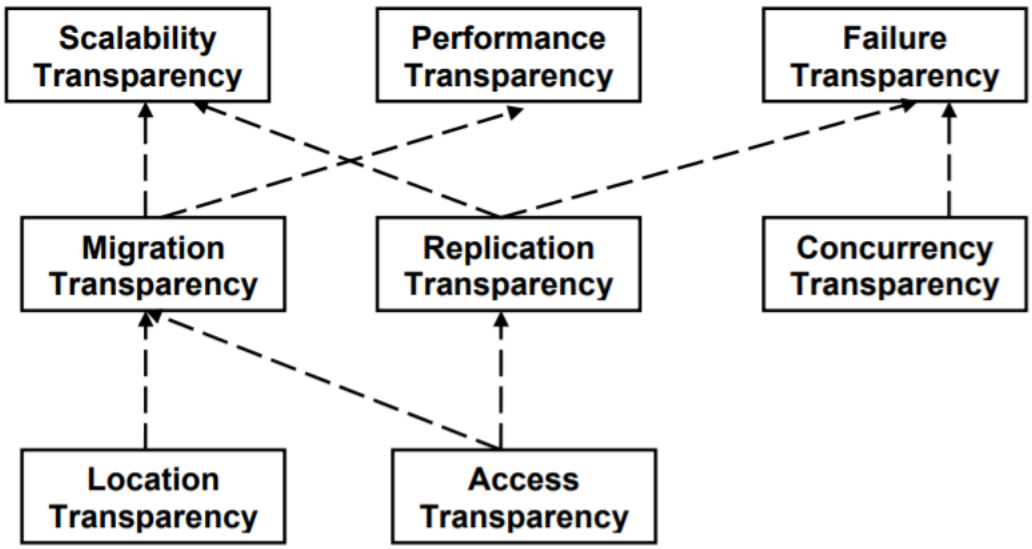


Figure 1.8: Transparency in distributed systems.

Access transparency: Enables local and remote information objects to be accessed using identical operations.

- Example: Navigation in the Web.
- Example: SQL Queries.

Location transparency: Enables information objects to be accessed without knowledge of their location.

- Example: Pages in the Web
- Example: Tables in distributed databases

Concurrency transparency: Enables several processes to operate concurrently using shared information objects without interference between them.

- Example: Automatic teller machine network.
- Example: Database management system.

Replication transparency: Enables multiple instances of information objects to be used to increase reliability and performance without knowledge of the replicas by users or application programs

- Example: Distributed DBMS.
- Example: Mirroring Web Pages.

Failure transparency: Enables the concealment of faults and Allows users and applications to complete their tasks despite the failure of other components.

- Example: Database Management System .

Migration transparency: Allows the movement of information objects within a system without affecting the operations of users or application programs.

- Example: NFS.
- Example: Web Pages.

Performance Transparency: Allows the system to be reconfigured to improve performance as loads vary.

- Example: Distributed make.

Scaling transparency: Allows the system and applications to expand in scale without change to the system structure or the application algorithms.

- Example: World-Wide-Web.
- Example: Distributed Database [8].

1.2.6 Advantages and disadvantages of distributed systems over centralized ones

Advantages:

- **Incremental growth**: Computing power can be added in small increments.
- **Reliability**: If one machine crashes, the system as a whole can still survive.
- **Speed** : A distributed system may have more total computing power than a mainframe.
- **Open system** : This is the most important point and the most characteristic point of a distributed system.
  - ✓ Since it is an open system it is always ready to communicate with other systems.

✓ An open system that scales has an advantage over a perfectly closed and self-contained system.

- **Economic** : And Microprocessors offer a better price/performance than mainframes.

Disadvantages:

- Complexity: They are more complex than centralized systems.
- Security: More susceptible to external attack.
- Manageability: More effort required for system management.
- Unpredictability: Unpredictable responses depending on the system organization and network load [9].

1.2.7 Distributed system architecture

In distributed architecture, components are presented on different platforms and several components can cooperate with one another over a communication network in order to achieve a specific objective or goal.

- In this architecture, information processing is not confined to a single machine rather it is distributed over several independent computers.
- A distributed system can be demonstrated by the client-server architecture which forms the base for multi-tier architectures; alternatives are the broker architecture such as CORBA, and the Service-Oriented Architecture (SOA).
- There are several technology frameworks to support distributed architectures, including .NET, J2EE, CORBA, .NET Web services, AXIS Java Web services, and Globus Grid services.
- Middleware is an infrastructure that appropriately supports the development and execution of distributed applications. It provides a buffer between the applications and the network. It sits in the middle of system and manages or supports the different components of a distributed system. Examples are transaction processing monitors, data convertors and communication controllers etc [10].

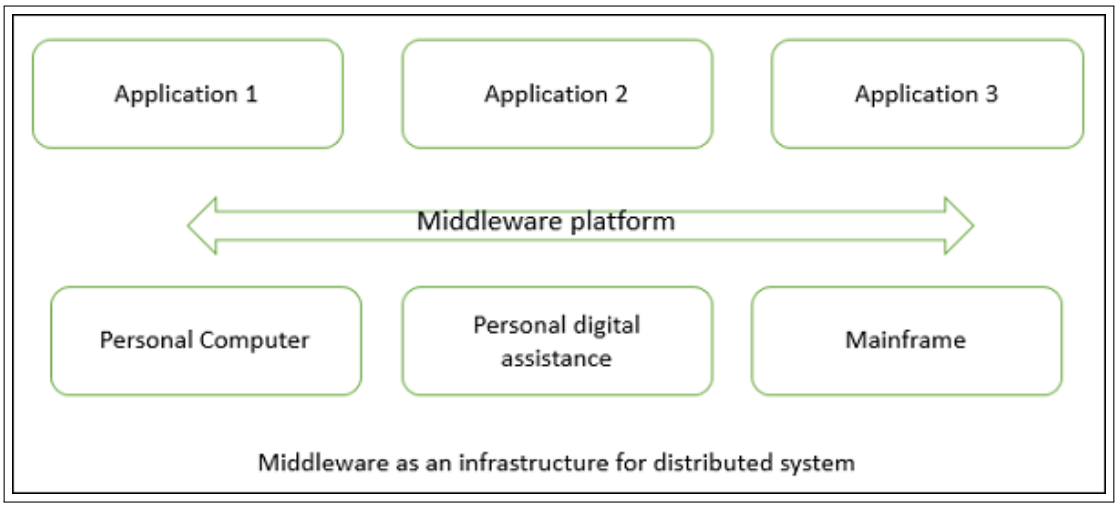


Figure 1.9: Middleware as an infrastructure for distributed system.

Client-server architecture

The client-server architecture is the most common distributed system architecture which decomposes the system into two major subsystems or logical processes

- **Client** This is the first process that issues a request to the second process i.e. the server.
- **Server** This is the second 9 process that receives the request, carries it out, and sends a reply to the client.

In this architecture, the application is modelled as a set of services that are provided by servers and a set of clients that use these services. The servers need not know about clients, but the clients must know the identity of servers, and the mapping of processors to processes is not necessarily.

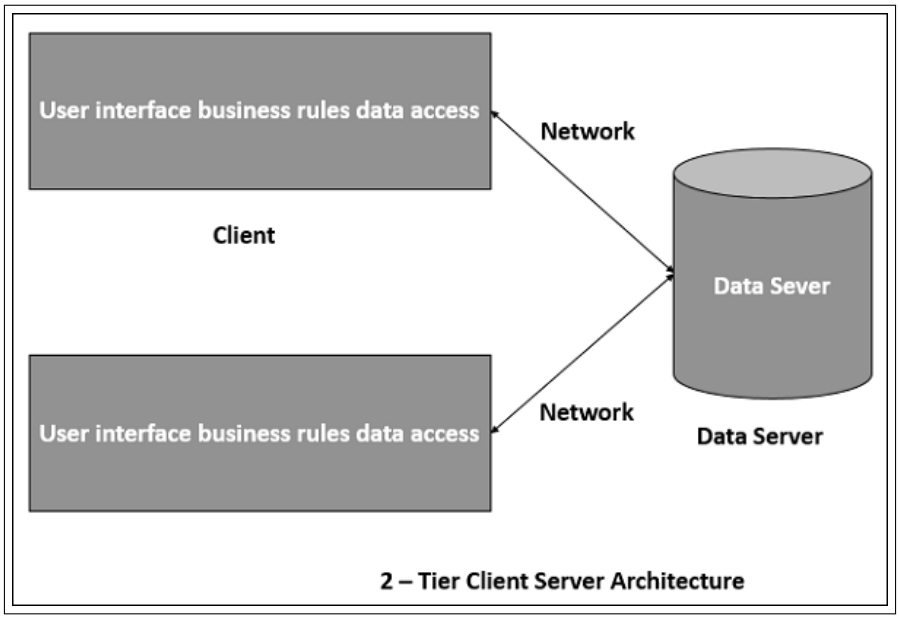


Figure 1.10: Client-server architecture.

Advantages And Disadvantages

Advantages:

- Separation of responsibilities such as user interface presentation and business logic processing.
- Reusability of server components and potential for concurrency.
- Simplifies the design and the development of distributed applications.
- It makes it easy to migrate or integrate existing applications into a distributed environment.
- It also makes effective use of resources when a large number of clients are accessing a high-performance server.

Disadvantages:

- Lack of heterogeneous infrastructure to deal with the requirement changes.
- Security complications.
- Limited server availability and reliability.
- Limited test-ability and scalability.
- Fat clients with presentation and business logic together [10].

Models architecture client-server

Client-server Architecture can be classified into two models based on the functionality of the client

Thin-client model:

In thin-client model, all the application processing and data management is carried by the server. The client is simply responsible for running the presentation software.

- Used when legacy systems are migrated to client server architectures in which legacy system acts as a server in its own right with a graphical interface implemented on a client
- A major disadvantage is that it places a heavy processing load on both the server and the network.

Thick/fat-client model:

In thick-client model, the server is only in charge for data management. The software on the client implements the application logic and the interactions with the system user.

- Most appropriate for new C/S systems where the capabilities of the client system are known in advance
- More complex than a thin client model especially for management. New versions of the application have to be installed on all clients.

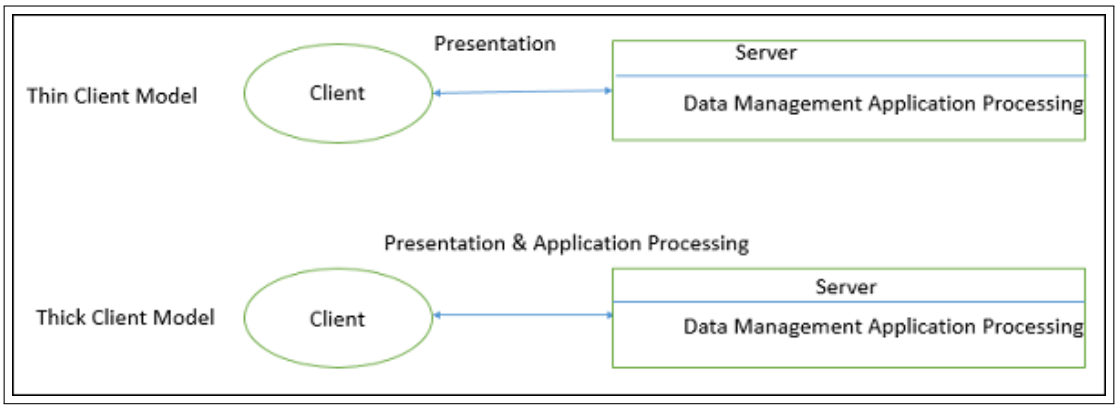


Figure 1.11: Thick/Fat-Client model.

Multi-tier architecture (n-tier architecture)

Multi-tier architecture is a clientserver architecture in which the functions such as presentation, application processing, and data management are physically separated. By separating an application into tiers, developers obtain the option of changing or adding a specific layer, instead of reworking the entire application. It provides a model by which developers can create flexible and reusable applications.

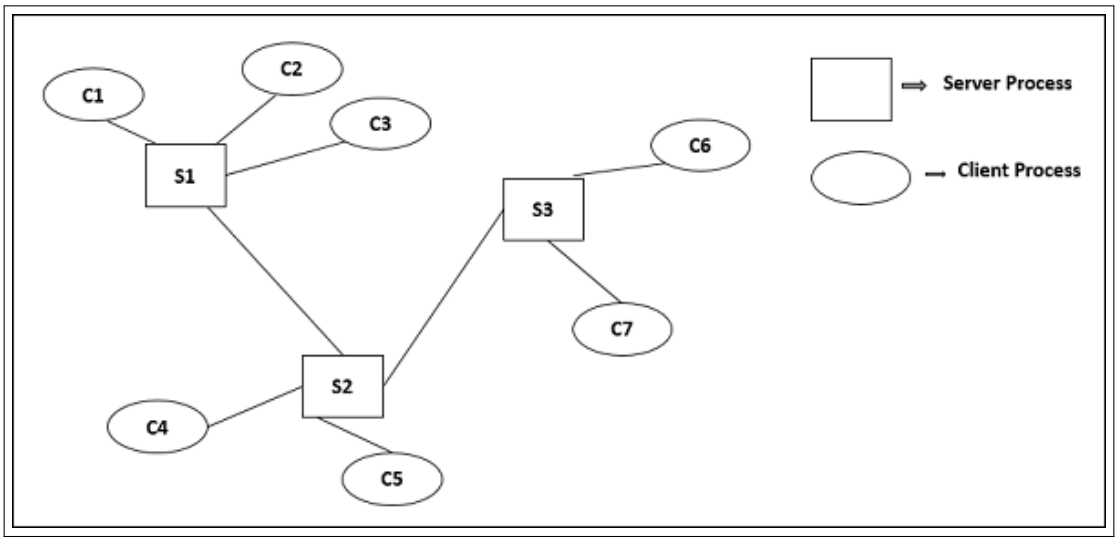


Figure 1.12: Thick/Fat-Client model.

Advantages and Disadvantages

Advantages:

- Better performance than a thin-client approach and is simpler to manage than a thick-client approach.
- Enhances the reusability and scalability as demands increase, extra servers can be added.
- Provides multi-threading support and also reduces network traffic.

- Provides maintainability and flexibility.

Disadvantages:

- Unsatisfactory Testability due to lack of testing tools.
- More critical server reliability and availability [10].

The three-tier architecture

The most general use of multi-tier architecture is the three-tier architecture. A three-tier architecture is typically composed of a presentation tier, an application tier, and a data storage tier and may execute on a separate processor.

Presentation tier: Presentation layer is the topmost level of the application by which users can access directly such as web page or Operating System GUI (Graphical User interface). The primary function of this layer is to translate the tasks and results to something that user can understand. It communicates with other tiers so that it places the results to the browser/client tier and all other tiers in the network.

Application tier (business logic, Logic Tier, or Middle Tier): Application tier coordinates the application, processes the commands, makes logical decisions, evaluation, and performs calculations. It controls an applications functionality by performing detailed processing. It also moves and processes data between the two surrounding layers.

Data tier: In this layer, information is stored and retrieved from the database or file system. The information is then passed back for processing and then back to the user. It includes the data persistence mechanisms (database servers, file shares, etc.) and provides API (Application Programming Interface) to the application tier which provides methods of managing the stored data.

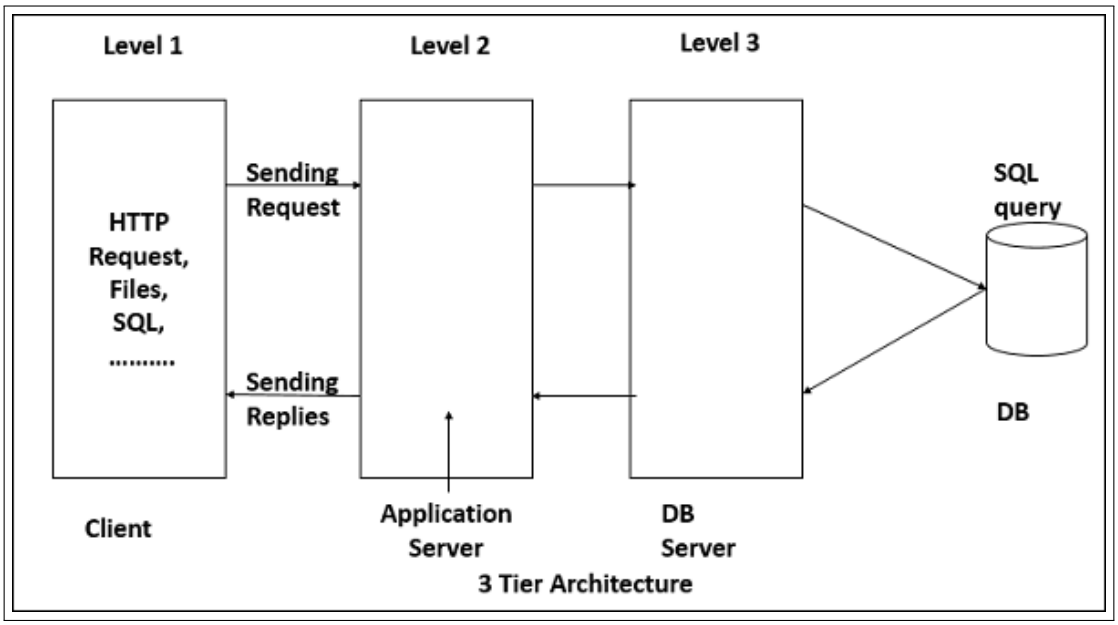


Figure 1.13: Architecture tier.

Broker architectural style

Broker Architectural Style is a middle-ware architecture used in distributed computing to coordinate and enable the communication between registered servers and clients. Here, object communication takes place through a middle-ware system called an object request broker (software bus).

- Client and the server do not interact with each other directly. Client and server have a direct connection to its proxy which communicates with the mediator-broker.
- A server provides services by registering and publishing their interfaces with the broker and clients can request the services from the broker statically or dynamically by look-up.
- CORBA (Common Object Request Broker Architecture) is a good implementation example of the broker architecture.

Components of broker architectural style

The components of broker architectural style are discussed through following heads

Broker: Broker is responsible for coordinating communication, such as forwarding and dispatching the results and exceptions. It can be either an invocation-oriented service, a document or message - oriented broker to which clients send a message.

- It is responsible for brokering the service requests, locating a proper server, transmitting requests, and sending responses back to clients.
- It retains the servers registration information including their functionality and services as well as location information.
- It provides APIs for clients to request, servers to respond, registering or unregistering server components, transferring messages, and locating servers.

Stub: Stubs are generated at the static compilation time and then deployed to the client side which is used as a proxy for the client. Client-side proxy acts as a mediator between the client and the broker and provides additional transparency between them and the client; a remote object appears like a local one.

The proxy hides the IPC (inter-process communication) at protocol level and performs marshaling of parameter values and un-marshaling of results from the server.

Skeleton: Skeleton is generated by the service interface compilation and then deployed to the server side, which is used as a proxy for the server. Server-side proxy encapsulates low-level system-specific networking functions and provides high-level APIs to mediate between the server and the broker.

It receives the requests, unpacks the requests, unmarshals the method arguments, calls the suitable service, and also marshals the result before sending it back to the client.

Bridge: A bridge can connect two different networks based on different communication protocols. It mediates different brokers including DCOM, .NET remote, and Java CORBA brokers.

Bridges are optional component, which hides the implementation details when two brokers inter-operate and take requests and parameters in one format and translate them to another format.

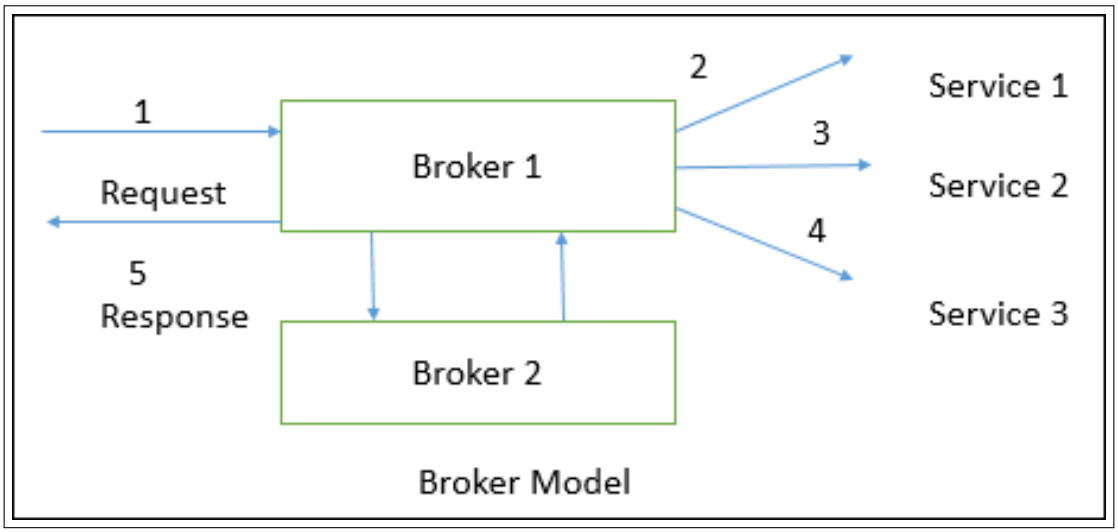


Figure 1.14: Architecture BROKER .

Broker implementation in CORBA: CORBA is an international standard for an Object Request Broker a middle ware to manage communications among distributed objects defined by OMG (object management group).

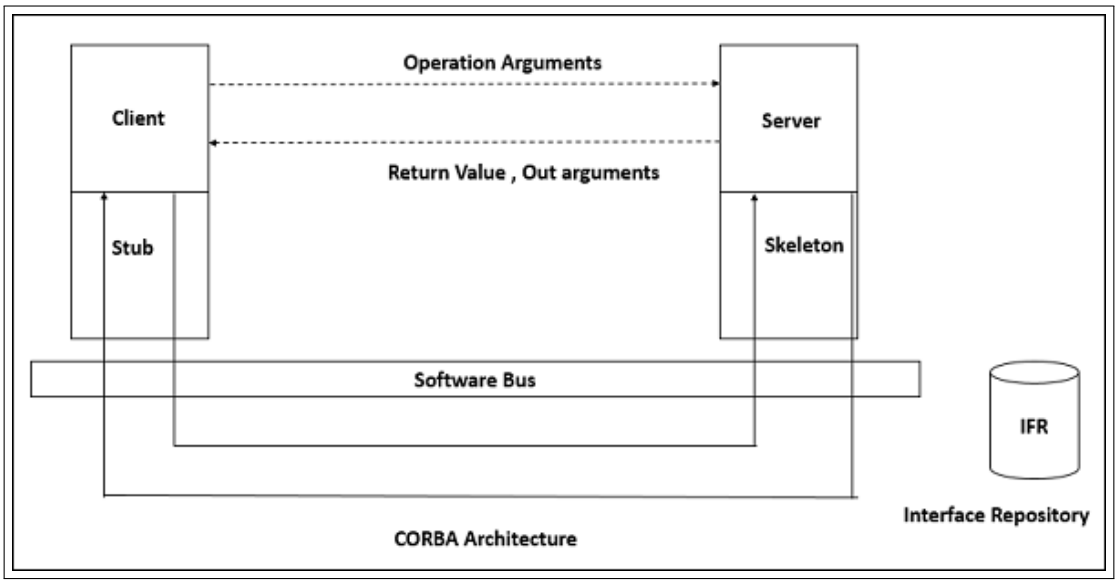


Figure 1.15: Architecture CORBA.

Service-oriented architecture (SOA)

A service is a component of business functionality that is well-defined, self-contained, independent, published, and available to be used via a standard programming interface. The connections between services are conducted by common and universal message-oriented protocols such as the SOAP Web service protocol, which can deliver requests and responses between services loosely.

Service-oriented architecture is a client/server design which support business-driven IT approach in which an application consists of software services and software service consumers (also known as clients or service requesters).

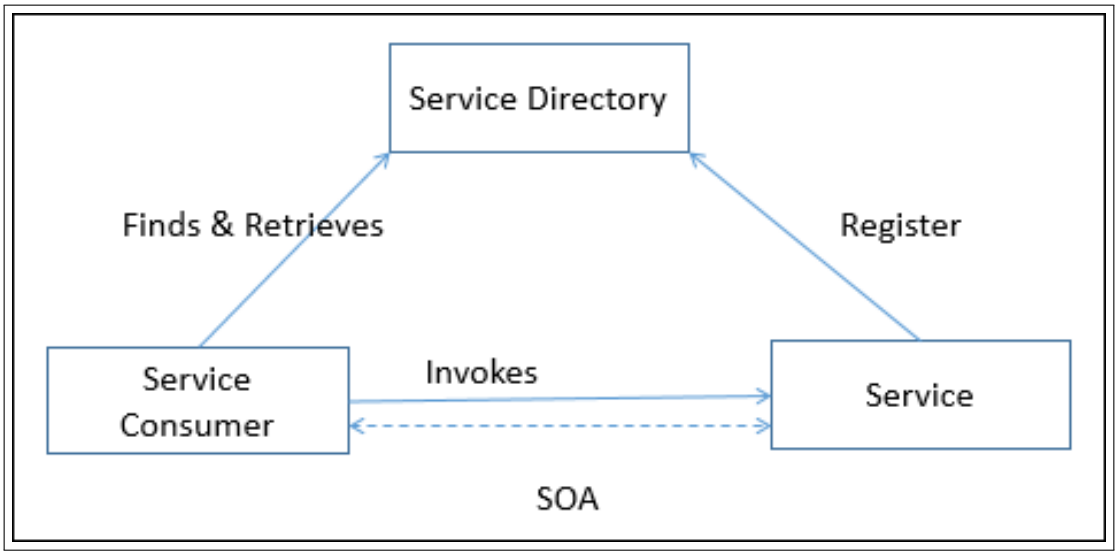


Figure 1.16: Architecture SOA.

Features of SOA: A service-oriented architecture provides the following features.

Distributed deployment: Expose enterprise data and business logic as loosely, coupled, discoverable, structured, standard-based, coarse-grained, stateless units of functionality called services.

Composability: Assemble new processes from existing services that are exposed at a desired granularity through well defined, published, and standard complaint interfaces.

Interoperability: Share capabilities and reuse shared services across a network irrespective of underlying protocols or implementation technology.

Reusability: Choose a service provider and access to existing resources exposed as services.

SOA Operation: The following figure illustrates how does SOA operate.

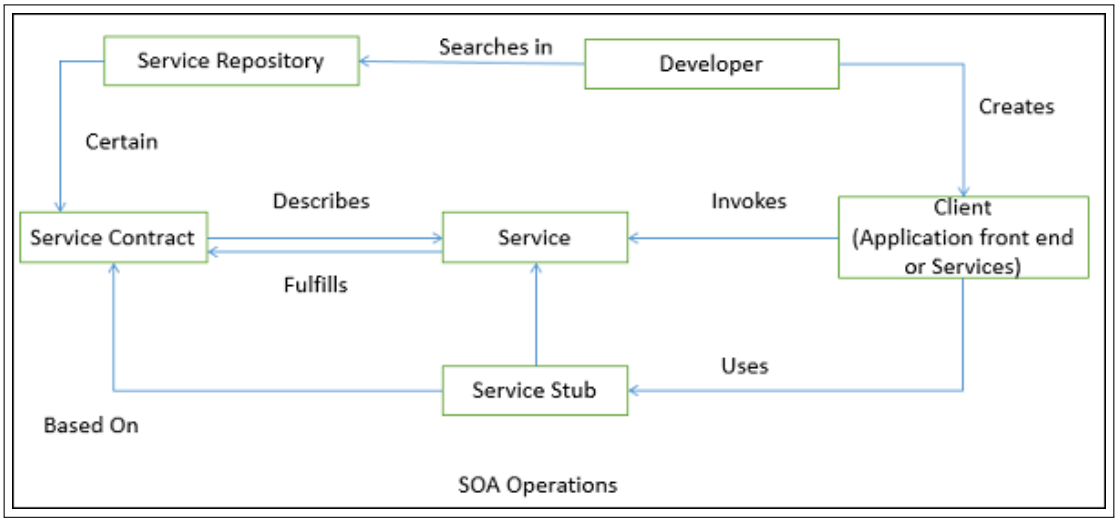


Figure 1.17: Architecture operation (SOA).

1.3 Complexity algorithms in distributed systems

1.3.1 Introduction

Large-scale distributed systems such as internet systems, ubiquitous computing environments, grid systems, storage systems, enterprise systems and sensor networks often contain immense numbers of heterogeneous and mobile nodes. These systems are highly dynamic and fault-prone as well.

The complexity of distributed computing systems for people has been widely identified to be an important problem. Many enterprises, governments and other organizations have large, complex computing systems that are difficult to manage, maintain and change. Application and service developers often face steep learning curves before they can start programming large distributed systems. Organizations also tend to spend huge amounts in administration and maintenance costs, which often exceed the cost of buying the system in the first place. End-users, too, are often overwhelmed with the complexity of a single computer, let alone multiple, heterogeneous devices.

In spite of the fact that system complexity has been widely talked about, the term complexity is often used loosely in connection with computer systems. There are no standard definitions of complexity or ways of measuring the complexity of large systems. As with so many complex things, computer system complexity means different things to different people [11].

1.3.2 Definition the complexity algorithm

The traditional definition of the complexity of an algorithm is in terms of its time and space requirements , That is, it is a measure of the amount of time and space required by an algorithm for an input of a given size (n). Or its relation to Turing machines or universal computers.

1.3.3 Cognitive aspect to algorithmic complexity

which is the effort required to understand an algorithm. There is often a trade-off between the performance and cognitive aspects of algorithmic complexity. Simple algorithms are often brute-force in nature and may have high space or time complexity. However, more sophisticated algorithms that reduce the space or time complexity have a high cognitive complexity.

A simple example is the use of an O(n2) algorithm for sorting as opposed to an O(n.log n) algorithm [11].

1.3.4 Relation between the complexity-flexibility

To reducing complexity through high-level programming and self-repair is that the flexibility of the developer is also reduced. While high-level and task-oriented programming abstracts away many of the low-level details, it is also not as expressive or flexible as lower-level programming and does not allow developers to perform certain kinds of operations. Flexibility is related to the number of

different states that a language or system allows to be reached, or the number of different transitions between states offered by operators of the language or system .The complexity-flexibility trade-off plays out at other levels of programming as well.

For example, machinelevel or assembly-level programming is probably the most expressive and flexible since it allows developers to do pretty much anything allowed by the instruction set of the processor. We also mention that Java is among the high-level and least complex programming languages, but also allow lesser flexibility. Fig. Below shows a possible graph representing the tradeoff between flexibility and complexity [11].

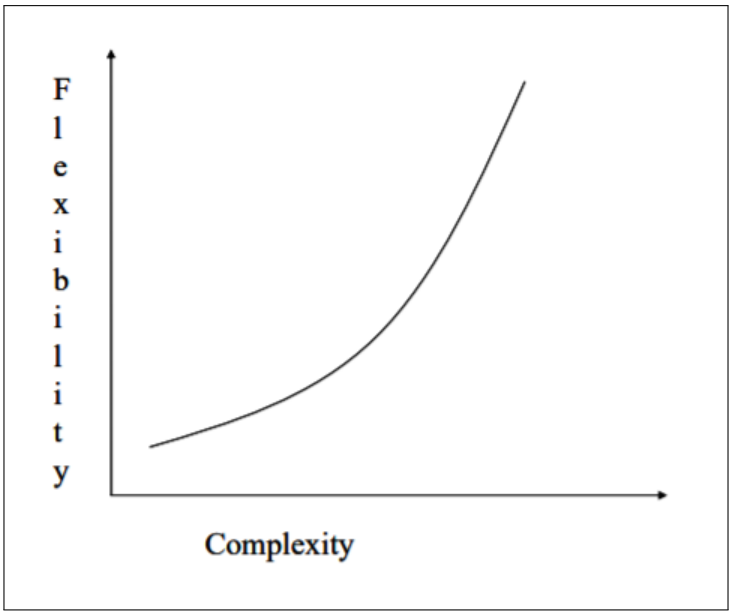


Figure 1.18: The complexity-flexibility tradeoff.

1.3.5 Different aspects of distributed system complexity

The term complexity has been widely used in different contexts by different people. In general, though, system complexity can be described as a measure of how understandable a system is and how difficult it is to perform tasks in the system. Understanding and using a system of high complexity requires great mental or cognitive effort, while a system of low complexity is easy to understand and use. In this section, we attempt to capture some of the aspects of systems that make them difficult to understand. Fig. 1 shows these aspects of complexity [11].

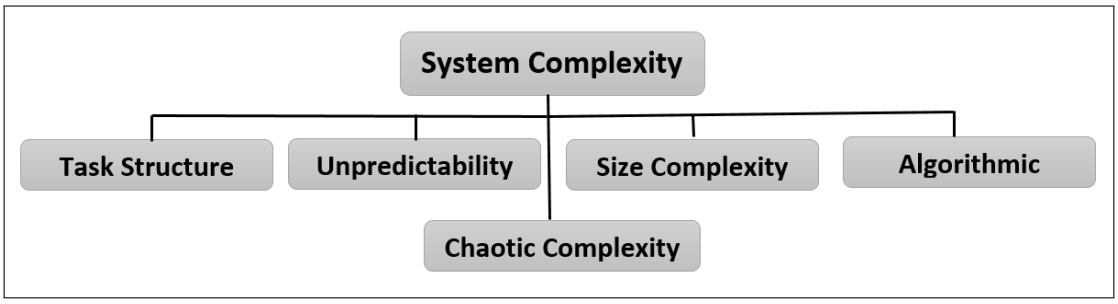


Figure 1.19: Different aspects of distributed system complexity.

1.3.6 Distinguishing between the complexity of a system and the complexity of using a system

There is an important difference between the intrinsic complexity of a system and the complexity of using a system to perform tasks. While a system can be very complex in the inside, it may still be easy to use for performing various kinds of tasks. system may be very complex, internally, it is possible to hide that complexity and present relatively simple interfaces to developers.

In general, the complexity of a system has to be measured with respect to a certain level of abstraction. The complexity of the same system may be different for an end-user, an administrator, an application developer, a middleware programmer or an operating system developer [11].

1.3.7 Conclusion

So far, there has not been much work in defining or measuring the complexity of distributed systems. Col yer et al have described the problem of middle ware complexity and have shown how aspect oriented software development can simplify development. Boo ch has also discussed why software is inherently complex and how abstractions and aspects can reduce complexity. They, however, do not propose quantitative metrics for measuring complexity.

1.4 General guidelines that can help reduce the complexity of systems

We propose various guidelines that can help reduce the different aspects of complexity. This is not meant to be an exhaustive set of solutions, but rather, examples of some of the approaches that can be taken to reduce system complexity [12].

1.4.1 Overcoming complexity

high-level programming and interaction

Cognitive algorithmic complexity can be reduced by using high-level programming. Any computer program consists of various operators and operands. In order to allow developers to perform high-level programming, we need middleware or frameworks that allow programming using high-level, abstract operators and operands. These high-level operators and operands are resolved by the framework or the middleware to appropriate low-level functions and operands either during compile-time or during run-time. Thus, the number of operators and operands used in a program will reduce, and hence the program volume and programming effort will also reduce. This will, thus, reduce cognitive algorithmic complexity [11].

Hierarchical organization of systems

Many large distributed systems are organized hierarchically, which allows dealing with smaller parts of the system independently. And Size complexity can be effectively tackled with a divide-and conquer approach.

However, there has not been much work in organizing the knowledge required to develop, manage and use distributed systems hierarchically. One way of doing this is to develop ontologies that define hierarchies of concepts used in the distributed system. Ontologies are a standard way of representing domain knowledge in a reusable manner. They allow different parties to become aware of the various concepts used in the system and the relationships between these concepts [11].

Configuration ,optimization and repair self

One way of reducing task-structure complexity is by building autonomic systems that can configure, optimize and repair themselves. This would allow the specification and execution of tasks to be more linear, since the system can take care of making choices at various decision points automatically. Hence, administrators and endusers do not have to worry about choosing appropriate system configuration and operation parameters. The system will automatically decide appropriate values to use at decision points and appropriate actions to take upon failures.

In distributed computing, many of the decision points that contribute to cyclomatic complexity arise from checking for exceptions and failures. If the distributed system provides middleware that performs selfconfiguration and self-repair, then programmers, too, do not have to deal with failures, exceptions and error conditions. Self-configuring systems also allow developers to focus on the logic of the program and leave many low-level parameters and properties to be taken care of by the system, automatically. Hence, many decision points in programs can be eliminated and the complexity of developing large distributed systems can be reduced [11].

CHAPTER 2

BLOCKCHAIN

2.1 Concepts

2.1.1 Introduction

As life increasingly moves online, one of the challenges Internet users face is conducting financial transactions in a setting where they cannot know or trust the other party. Some of these trust issues have been alleviated through the development of cryptocurrencies like Bitcoin. All transactions in the Bitcoin economy are tracked in a ledger system called the blockchain,Blockchain is an open, distributed ledger that records transactions between parties efficiently and in a verifiable and permanent manner.

Multiple copies of this ledger exist and are constantly compared to each other to ensure that all transactions are legitimate and properly recorded. There are many other cryptocurrencies similar to Bitcoin, but all use a similar type of blockchain ledger verification system. While cryptocurrencies are an interesting and important topic, the blockchain ledger itself has many other potential uses outside of currency markets, including creating tamper-proof documents, distributed ownership records, and universal medical records.

2.1.2 The history of blockchain

The underlying technology behind cryptocurrencies is the blockchain. It allows every client in the network to reach consensus without ever having to trust each other.

The early days. The blockchain technology was described in 1991 by the research scientist Stuart Haber and W. Scott Stornetta. They wanted to introduce a computationally practical solution for time-stamping digital documents so that they could not be backdated or tampered.

The system used a cryptographically secured chain of blocks to store the time-stamped documents And In 1992, Merkle Trees were incorporated into the design, which makes blockchain more efficient by allowing several documents to be collected into one block. Merkle Trees are used to create a 'secured chain of blocks.' It stored a series of data records, and each data records connected to the one before it. The newest record in this chain contains the history of the entire chain. making it more efficient by allowing several documents to be collected into one block. However, this technology went unused and the patent lapsed in 2004, four years before the inception of Bitcoin [13].

2.1.3 Blockchain defined

Layman's definition: Blockchain is an ever-growing, secure, shared record keeping system in which each user of the data holds a copy of the records, which can only be updated if all parties involved in a transaction agree to update.

Technical definition: Blockchain is a peer-to-peer, distributed ledger that is cryptographically-secure, append-only, immutable (extremely hard to change), and updateable only via consensus or agreement among peers [14].

2.1.4 The concept of blockchain

Blockchain technologies is not just only single one technique, but contains Cryptography, mathematics, Algorithm and economic model, combining peer-to-peer networks and using distributed consensus algorithm to solve traditional distributed database synchronize problem, its an integrated multi-field infrastructure construction.

The blockchain technologies composed of six key elements.

Decentralized: The basic feature of blockchain, means that blockchain doesnt have to rely on centralized node anymore, the data can be record, store and update distributedly.

Transparent: The datas record by blockchain system is transparent to each node, it also transparent on update the data, that is why blockchain can be trusted.

Open source: Most blockchain system is open to everyone, record can be check publicly and people can also use blockchain technologies to create any application they want.

Autonomy: Because of the base of consensus, every node on the blockchain system can transfer or update data safely, the idea is to trust form single person to the whole system, and no one can intervene it.

Immutable: Any records will be reserved forever, and cant be changed unless someone can take control more than 51 percent node in the same time.

Anonymity: Blockchain technologies solved the trust problem between node to node, so data transfer or even transaction can be anonymous, only need to know the persons blockchain address [15].

How blockchain works ?

The main working processes of blockchain are as follows:

- The sending node records new data and broad casting to network.
- The receiving node checked the message from those data which it received, if the message was correct then it will be stored to a block.
- All receiving node in the network execute proof of work (PoW) or proof of stake (PoS) algorithm to the block.
- The block will be stored into the chain after executing consensus algorithm, every node in the network admit this block and will continuously extend the chain base on this block [15].

The Structure of blockchain

Generally in the block, it contains main data, hash of previous block, hash of current block, timestamp and other information.

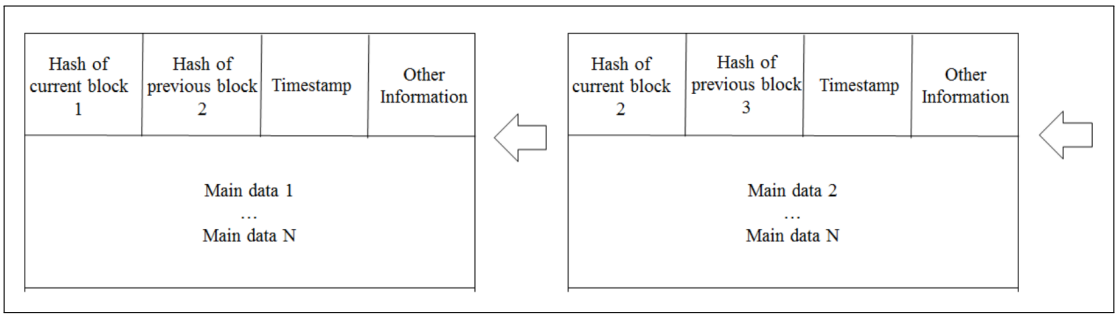


Figure 2.1: Shows the structure of block.

Main data: Depending on what service is this blockchain applicate, for example: transaction records, bank clearing records, contract records or IOT data record.

Hash: When a transaction executed, it had been hash to a code and then broadcast to each node. Because it could be contained thousands of transaction records in each nodes block, blockchain used Merkle tree function to generate a final hash value, which is also Merkle tree root. This final hash value will be record in block header (hash of current block), by using Merkle tree function, data transmission and computing resources can be drastically reduced.

Timestamp: Time of block generated.

Other Information: Like signature of the block, Nonce value, or other data that user define [15].

Type of blockchain

The following table represents the differentiation among all kind of blockchains.

BC/ Properties	Efficiency	Decentralized	Accord growth	immovableness	Reading	Determining
Private BC	good	No	yes	Can be	Can be publicly	Only one industry
Public BC	worse	Yes	no	No	publicly	All miners
Consortium BC	good	Some times	yes	Can be	Can be publicly	IoT devices

Figure 2.2: Kinds of blockchains and their properties.

Blockchain technologies can be roughly divided into three types.

Public blockchain: Everyone can check the transaction and verify it, and can also participate the process of getting consensus. Like Bitcoin and Ethereum are both Public Blockchain.

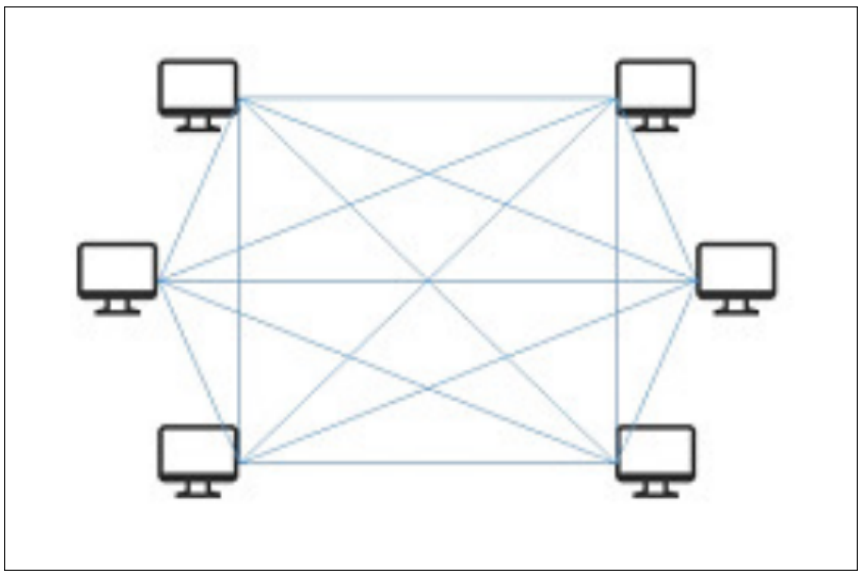


Figure 2.3: Shows public blockchain.

Consortium blockchains: It means the node that had authority can be choose in advance, usually has partnerships like business to business, the data in blockchain can be open or private, can be seen as Partly Decentralized. Like Hyperledger and R3CEV are both consortium blockchains.

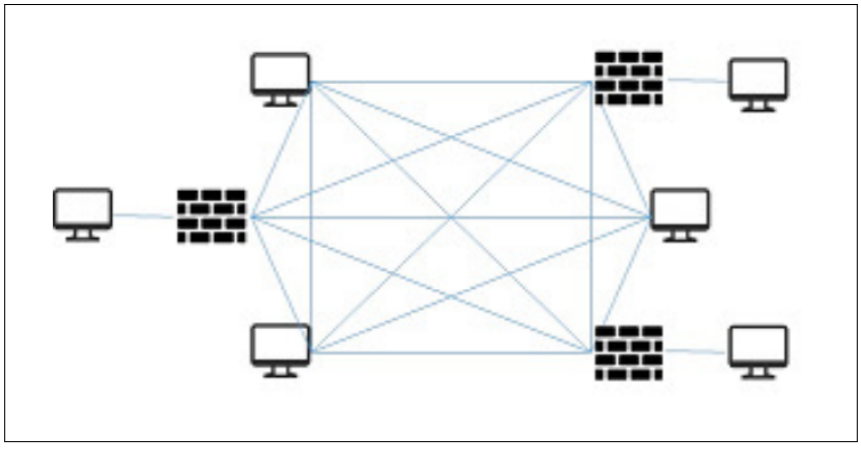


Figure 2.4: Shows consortium blockchains.

Private blockchain: Node will be restricted, not every node can participate this blockchain, has strict authority management on data access. No matter what types of blockchain is, it both has advantage. Sometimes we need public blockchain because its convenience, but sometimes we maybe need private control like consortium blockchains or private blockchain, depending on what service we offer or what place we use it [15].

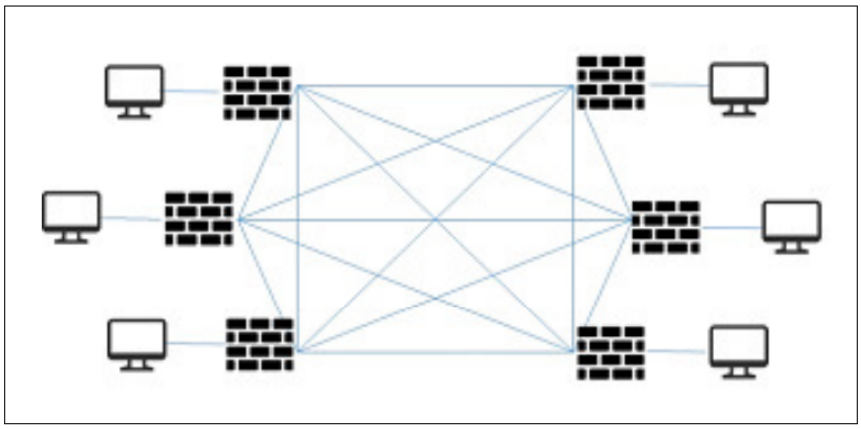


Figure 2.5: Shows private blockchain.

2.1.5 Advantages and Disadvantages of blockchain

Advantages

Process integrity: Due to the security reasons, this program was made in such a way that any block or even a transaction that adds to the chain cannot be edited which ultimately provides a very high range of security.

Traceability: The format of Blockchain designs in such a way that it can easily locate any problem and correct if there is any. It also creates an irreversible audit trail.

Security: Blockchain technology is highly secure because of the reason each and every individual who enters into the Blockchain network is provided with a unique identity which is linked to his account. This ensures that the owner of the account himself is operating the transactions. The block encryption in the chain makes it tougher for any hacker to disturb the traditional setup of the chain [16].

Disadvantages

Power use: The consumption of power in the Blockchain is comparatively high as in a particular year the power consumption of Bitcoin miners was alone more than the per capita power consumption of 159 individual countries. Keeping a real-time ledger is one of the reasons for this consumption because every time it creates a new node, it communicates with each and every other node at the same time.

Cost: As per the studies as an average cost of the Bitcoin transaction is $75-160$ and most of this cost cover by the energy consumption. There are very fewer chances that this issue we can resolve by the advancement in the technology. As the other factor that is the storage problem might be covered by the energy issues cannot be resolved.

Uncertain regulatory status: In each and every part of world modern money has been created and controlled by the central government. It becomes a hurdle for Bitcoin to get accepted by the

preexisting financial institutions [16].

2.2 Problematic

2.2.1 Introduction

In 1982, a thought experiment was proposed by Lamport and others in their research paper, The Byzantine Generals Problem which is available whereby a group of army generals who lead dif erent parts of the Byzantine army are planning to attack or retreat from a city. The only way of communicating among them is via a messenger. They need to agree to strike at the same time in order to win. The issue is that one or more generals might be traitors who could send a misleading message. So what is the solution to get out of this problem.

2.2.2 The issue of the problem of the byzantine generals

there is a need for a viable mechanism that allows for agreement among the generals, even in the presence of the treacherous ones, so that the attack can still take place at the same time. As an analogy to distributed systems, the generals can be considered nodes, the traitors as Byzantine (malicious) nodes, and the messenger can be thought of as a channel of communication among the generals . This problem was solved in 1999 by Castro and Liskov who presented the Practical Byzantine Fault Tolerance (PBFT) algorithm, where consensus is reached after a certain number of messages are received containing the same signed content.

Figure 2.6: Problem of the byzantine generals.

2.2.3 What is consensus algorithm

A consensus algorithm may be defined as the mechanism through which a blockchain network reach agreement and make decisions. And all nodes should agree about the changes in the distributed

ledger - Proof-of-Work (PoW) ,Proof-of-Stake (PoS),and other algorithms. Since Blockchain network does not rely on a central authority, the distributed nodes need to agree on the validity of transactions.

Proof-of-Work

What is Proof-of-Work ?

Proof-of-Work, or PoW, is the original consensus algorithm in a Blockchain network. and is a protocol that works by exhausting the resources of a computer system to prevent hacking and attacks ,Originated in 1993 by Cynthia Dwork and Moni Naor.

Principle of proof of work

Proof-of-Work (PoW) is one of several common approaches to reach a consensus in the Blockchain. A Proof-of-Work (PoW) is the original consensus algorithm in a blockchain network a cryptographic puzzle that is difficult to solve but easy to verify. PoW's difficulty can be adjusted to reflect the long-time growth of a Blockchain and improved technical means. Currently, taking the Bitcoin operation as real-world case of Blockchain use, the block rate is about one block in 10 minutes . The hash value for the block header in PoW is calculated for each network node. The header of the block contains a nonce and miners frequently modified the nonce to have values for different hash. The consensus involves the calculated value to be the same or less the value. When a node reaches the target, the block is announced to all other nodes, and entire other nodes confirm the hash's accuracy together. When the block has been validated ,other miners attach it to their own blockchains [17].

Advantages of proof of work

- Algorithm selection such as Monero to resistant ASIC to thwart these drifts.
- Proof of work effectively secures the network by making hacking attempts very difficult.
- the more participants the network has, the more secure the network
- There are mainly two features that have contributed to the wide popularity of this consensus protocol and they are:
  - ✓ It is hard to find a solution for the mathematical problem
  - ✓ It is easy to verify the correctness of that solution [18].

Disadvantages of proof of work

- Proof of Work in some cases poses energy and environmental issues, requiring sometimes drastic energy consumption.
- The switch to ASIC also poses a major centralization problem, leaving the field open to mining farms and other big players to grab most of the network computing power, and concentrate the network in the hands of a small number of actors [18].

Main issues with the proof-of-work consensus:

The Proof-of-Work consensus mechanism has some issues which are as follows:

- **The 51 % risk:** If a controlling entity owns or more than 51 % of nodes in the network, the entity can corrupt the blockchain by gaining the majority of the network (double-spend money) deny service ů
- **Resource consumption:** (money, energy, space, hardware).
- **Time consuming:** Its operations which is a time consuming , Transaction confirmation takes about 1060 minutes. So, it is not an instantaneous transaction; because it takes some time to mine the transaction and add it to the blockchain thus committing the transaction [19].

Other alternatives to the PoW consensus:

These issues have led to the development of new consensus protocols like:

- Proof of stake
- Proof of burn
- Raft consensus
- Proof of activity
- Proof of elapsed time
- Proof of capacity
- Proof of importance

Proof of stake

Introduction

l have PoS finds direct applicability in open blockchain platforms and has been seen as a strong candidate to replace the largely inefficient Proof of Work mechanism that is currently plugged in most existing open blockchains. And also to solve the energy expenditure problem created by PoW [20].

What is proof of stake ?

Proof of stake is a type of consensus mechanism that blockchain networks use to achieve distributed consensus. It includes choosing the next block creator by different combinations of random selection, fortune, lifetime of coins or tokens [21].

The principle of use Proof of Stake (PoS) ?

PoS attempts to solve the problem of energy wastage caused by PoW. To do this, it replaces PoS that competes with PoW by randomly selecting stakeholders to attach to the blockchain. The simplest implementation of PoS involves each blockchain branch selecting uniformly and randomly from the universe of blockchain coins. The owner of the selected coin then receives the option to append to the

branch that selected her coin and simultaneously collect a reward. This protocol succeeds in reducing energy expenditure to negligible levels [22].

Disadvantages of proof of stake

- PoS threatens to recreate an issue ostensibly solved by PoW, which is nothing at stake.
- most existing PoS variants suffer from the nothing at stake and the long range attacks which considerably degrade security in the blockchain.

Proof-of-stake VS Proof-of-work

As opposed to proof-of-work , proof-of-stake requires nodes to have a certain amount of tokens in order to qualify to validate blocks. In a basic PoS-enabled system, the more of a certain cryptocurrency, token a miner owns be it Bitcoin, Ether, the more likely that person is to be chosen to mine blocks. Hence, the more power that miner has in the system.

Proof of burn

What is a proof of burn ?

Proof of burn is one of the several consensus mechanism algorithms implemented by a blockchain network to ensure that all participating nodes come to an agreement about the true and valid state of the blockchain network. It is one of the alternative consensus algorithms that attempt to address the problem of high system power consumption. It is compared to Proof of Work (PoW) [23].

Raft consensus

What is raft consensus ?

Raft is a consensus algorithm that is designed to be easy to understand. And in order to better understand how to consensus, It's equivalent to Paxos in fault-tolerance and performance. It was developed by Diego Ungaro and John Oosterhout [24].

Proof-of-activity

What is Proof-of-Activity (PoA) ?

Proof-of-Activity is defined as a blockchain consensus algorithm that doesn't only ensure that transactions are genuine but also ensures that miners reach a consensus, It is used to ensure that all transactions occurring on the blockchain are genuine, PoA is a combination of two other blockchain consensus algorithms: Proof-of-Work (PoW) and Proof-of-Stake (PoS) [25].

Proof of elapsed time

What is Proof of Elapsed Time (PoET) ?

Proof of elapsed time (PoET) is a blockchain network consensus mechanism algorithm that prevents high resource utilization and high energy consumption ,And that is often used on the permissioned blockchain networks to decide the mining rights or the block winners on the network By running a trusted code within a secure environment, the PoET mechanism is based on spreading the chances of a winning fairly across the largest possible number of network participants. Proof of Elapsed Time (PoET) is an efficient alternative to proof of work (PoW) [26].

Proof of capacity

What is Proof of Capacity (PoC) ?

Proof of capacity is a consensus mechanism algorithm used in blockchains that allows the mining devices in the network to use their available hard-drive space to perform the mining process. This is in contrast to using the mining devices computational power as in the proof of work algorithm or the miners stake in the cryptocurrencies as in the proof of stake algorithm.

This approach provides a much cheaper and greener, thus more efficient and viable source of block verification [27].

Proof of importance

What is Proof of Importance (PoI) ?

Proof-of-Importance (PoI) is a blockchain consensus mechanism that was first introduced by NEM.it is system used to determine which users are eligible to perform the calculations necessary to add a new block of data to a blockchain and receive the associated payment. PoI reward users that actively transact on the network, building on proof-of-stake [28].

2.2.4 Why improvement in proof of work and not use other consensus algorithms

Proof of Work system is one that forces computers to do a little extra work before a requested process is executed. The extra work results in a solution, which is then presented to the other computers in a network. The other computers can easily verify that the solution is accurate and approve whatever action the original computer is requesting,This is a key characteristic of a Proof of Work system: it must be somewhat difficult to find a solution but extremely easy to verify that a particular There is a major benefit to Proof of Work. Proof of Work systems can be used to provide security to an entire network. This is the primary benefit for blockchains that use a Proof of Work consensus mechanism. If enough nodes (computers or dedicated mining machines) are competing to

find a specific solution, then the computational power needed to overpower and manipulate a network becomes unattainable for any single bad actor or even a single group of bad actors.

On the other hand, the restrictions that the proof of stake placed on the contract She can check a percentage of the transactions Since nodes can only validate a percentage of transactions equal to the percentage of cryptoassets they hold, it is unnecessary in many cases for them to continuously leverage computational power (and electricity) to compete for a block reward. This makes proof of stake largely more efficient and cost-effective than a proof of work model. Proof of work is notoriously inefficient, demanding excessive consumption of energy as well as a significant cost to miners.

In the last we proposed an algorithm (protocol) that improves Proof of Work (Reduce energy consumption) And we proved this through a group results of different experiments.

CHAPTER 3

BLOCKCHAIN APPLICATIONS

3.1 Introduction

With the popularity of blockchain, many companies, institutions and individuals have started using blockchain technology, because Blockchain has the potential to bring so much to the table. Although initially considered suitable for banking sectors only, blockchain has occupied many other sectors, and you can use it, for example, in the supply chain, healthcare, government, insurance, banking, real estate, and many more.

3.2 Blockchain applications

3.2.1 Blockchain application to cryptocurrencies

What is cryptocurrencys ?

A cryptocurrency is a digital currency that can be used to purchase goods and services online, which uses a ledger with strong cryptography to secure transactions. Cryptocurrencies operate using a technology called blockchain, and they are more than 6,700 different cryptocurrencies overall being the most popular Ethereum (ETH), Litecoin (LTC), Cardano (ADA), Polkadot (DOT), Bitcoin Cash (BCH), Stellar (XLM), Chain link, Binance. (BNB), Tether (USDT), Bitcoin is longer is the most popular digital currency [29].

Figure 3.1: Cryptocurrencys.

Advantages of the cryptocurrencies

- No inflation the maximum number of coins is strictly limited. Since there are neither political forces nor corporations that can change this order, there is no possibility of developing inflation in the system.

- There is no master server responsible for all operations in the cryptocurrency network as it works peer-to-peer.
- Unlimited transactionseach of the wallet holders can pay to everyone, anywhere and any amount. The transaction cannot be controlled or prevented, so you can make transfers anywhere in the world wherever a user is placed with a wallet.
- Payments made in this system are impossible for cancelation. Coins cannot be forged, copied or spent twice. These opportunities guarantee the integrity of the field system.
- There is no central controlling authority in the network, the network is alluded to all participants, each computer crypto-valued member is a member of this system.
- In cryptocurrencies government has no power to dictate rules to cryptocurrency owners. And even if some part of the network goes offline, the payment system will continue to function steadily [30].

Blockchain consensus mechanisms and Their relationship with cryptocurrencies

In principle, any node within a blockchain network can propose the addition of new information to the blockchain. In order to validate whether this addition of information is legitimate, the nodes have to reach some form of agreement. Here a consensus mechanism comes into play. In short, a consensus mechanism is a predefined specific (cryptographic) validation method that ensures a correct sequencing of transactions on the blockchain. In the case of cryptocurrencies, such sequencing is required to address the issue of double-spending (i.e. the issue that one and the same payment instrument or asset can be transferred more than once if transfers are not registered and controlled centrally).A consensus mechanism can be structured in a number of ways. The most common examples of consensus mechanisms in the context of cryptocurrencies are the Proof of Work (PoW) and the Proof of Stake (PoS) mechanism [31].

Cryptocurrency (Bitcoin)

What is bitcoin ?

Bitcoin is a digital currency that was created in January 2009. usually described as a virtual, decentralized and anonymous currency that is not government-backed or backed by any other legal entity It follows the ideas set out in a white paper by the mysterious and pseudonymous Satoshi Nakamoto. It is a currency that cannot be exchanged for gold or any other commodity [32].

How does bitcoin work ?

Anyone can use bitcoins to make payments to others without involving a third party, such as a bank or financial institution, for the purpose of verification. Instead, transactions are scanned and validated within the order by the blockchain [33].

characteristics bitcoin

To know more about Bitcoin, its history and work cannot be ignored and get acquainted with its characteristics.

- bitcoins are highly liquid, have low transaction costs, can be used to send payments quickly across the internet, and can be used to make micropayments.
- This new currency could also hold the key to allowing organizations such as Wikileaks, hated by governments, to receive donations and conduct business anonymously . There are basic characteristics of Bitcoin that make it different from a traditional currency.

Decentralization: The first and most important feature of Bitcoin is decentralization. There is no central power in Bitcoin as there is in the non-cryptocurrency traded currencies, which are issued and managed by a central authority. Bitcoin decentralization offers many advantages over traditional currency such as freedom from forfeiture.

Transparency: It is known to us that it is not possible to know how much bitcoins a person owns, but at the same time, it is clear to everyone on the ledger board that the amount of the transaction was made by any user and is the Bitcoin receiver. So, its treatment is pretty straightforward to everyone in the bitcoin ecosystem. From this date mentioned on the ledger board, based on a suitable analysis, the asset owned by anyone can easily be known if one so desires. But many things can be done to prevent this as well.

Opaquenes: The bitcoin user remains anonymous, and there is no opportunity to track the user. There are no requirements for any legal paper that helps establish a person's identity. This is also the reason why no government can know who is behind a particular account. At the same time, when you create an account with the bank or conduct transactions through the bank, they will ask you for personal information, and in transactions, they will have the details that you made from the delivery, receipt and withdrawal of amounts and the date of that as well.

Fast: Compared to other banks or any other method of transactions, bitcoin is faster, if the amount is sent through any bank or any other method of transactions , it will take almost a week or more. The world only takes a few minutes if it is sent in Bitcoin form.

Non-Repudiable: What comes under this characteristic is if bitcoin is transacted once, there is no getting back unless the receiver is willing to do so. It means there is no going back; the receiver cant claim that he never received any bitcoin.

Digital currency: Bitcoins are not physically present in the form of notes or coins, And in this way, it is easy to carry in the phone. It is tough to be stolen by thieves [34].

The main differences between bitcoin and blockchain:

- Bitcoin is a cryptocurrency, while blockchain is a distributed database.
- Bitcoin promotes anonymity, while blockchain is about transparency.
- Bitcoin transfers currency between users, while blockchain can be used to transfer all sorts of things, including information or property ownership rights.
- Bitcoin is powered by blockchain technology, but blockchain has found many uses beyond Bitcoin [35].

How are bitcoins stored and transactions made?

In order to safely store, send and receive bitcoins, individuals must setup a digital wallet

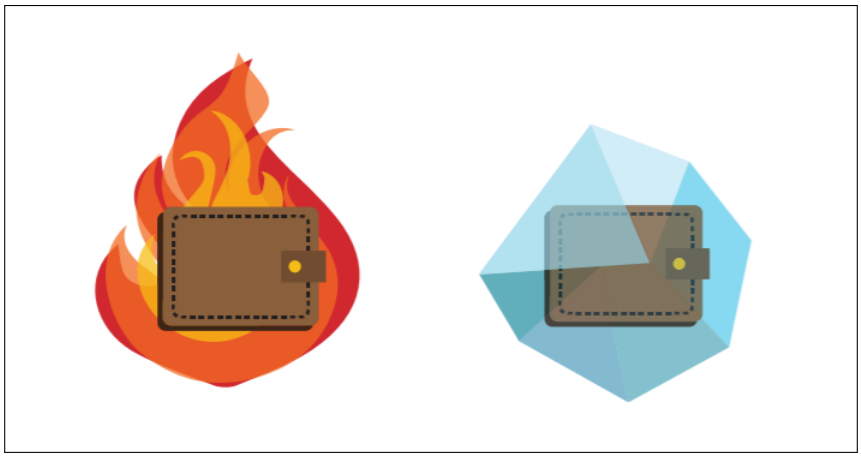


Figure 3.2: How to store bitcoins.

A wallet is a file that houses all the unique keys (passwords) attributed to an owners bitcoins and where their value will be recorded and synchronized when sending or receiving payments within the system. The digital wallet can be hardware-based or web-based. The wallet can also reside on a mobile device, on a computer desktop, or kept safe by printing the private keys and addresses used for access on paper. Wallets typically come in two forms: **hot** and **cold**. **Hot wallets** are connected to the internet and allow for instantaneous transfers online through the blockchain. **Cold wallets** are offline wallets that live on a users computer. Cold wallets offer a greater level of security but the transaction process is longer and more cumbersome compared to a hot wallet. In order to receive bitcoins, The process is to provide the sender with the address. The address is a long string of numbers and letters beginning with the number "1" or "3", which is the unique identifier used to send payments to the recipient. Only the sender can create new addresses for each transaction to maintain greater financial privacy [36].

Advantages of using bitcoin

- Safeguards Against Currency Manipulation ,Bitcoin is not owned or controlled by a country or governing body.
- Greater Consumer Protections ,The use of bitcoin as an alternative to fiat currency provides protection against the downside that can occur with traditional bank accounts.
- Greater Transparency ,Because all bitcoin transactions are permanently recorded on the blockchain, all transactions are public and traceable. The balance associated with each address is also part of the public record. The blockchain makes bitcoin much more transparent than many other monetary systems.
- Greater Liquidity Relative to Other Cryptocurrencies, As the most popular cryptocurrency by a significant margin, Bitcoin has far greater liquidity than its peers.
- Private and Secure, Although all bitcoin transaction details are stored publicly on the blockchain, the identities of the users involved remain relatively anonymous.[37]

Disadvantages of using bitcoin

- Exposure to Bitcoin-Specific Scams and Fraud : As the worlds most popular cryptocurrency, Bitcoin has seen more than its fair share of medium-specific scams, fraud, and attacks.
- Black Market Activity May Damage Reputation and Usefulness : Despite high-visibility prosecutions of the most egregious offenders, Bitcoin remains attractive to criminals and gray market participants.
- Awareness and Understanding: There is still very little awareness and knowledge about digital currencies. Not that many people have heard of Bitcoin.
- Susceptible to High Price Volatility and Instability :Although Bitcoin is the most liquid and easily exchanged cryptocurrency, it remains susceptible to wild price swings over short periods of time. Also, there is volatility in Bitcoin due to the limited amount of Bitcoin available and the increasing demand for it daily.
- No Chargebacks or Refunds: One of Bitcoin's biggest drawbacks is the lack of a uniform policy for refunds or refunds, as it is with traditional online payment processors. This prevents users from fraudulent transactions [38].

Cryptocurrency (Ethereum)

What is Ethereum ?

Ethereum is an open blockchain platform that lets anyone build and use decentralized applications that run on blockchain technology. launched in July 2015, Nobody controls or owns it is an open-source project built by many people around the world. But unlike the Bitcoin protocol, Ethereum was

designed to be adaptable and flexible. It is easy to create new applications on the Ethereum platform. and with the Homestead release, it is now safe for anyone to use those applications.

3.2.2 Blockchain and its role in IoT

Blockchain (BC) in the Internet of Things (IoT) is a novel technology that acts with decentralized, distributed, public and real-time ledger to store transactions among IoT nodes.

The transactions: in BC are the basic units that are used to transfer data between IoT nodes.

The IoT: nodes are different kind of physical but smart devices with embedded sensors, actuators, programs and able to communicate with other IoT nodes.

The role of BC in IoT is to provide a procedure to process secured records of data through IoT nodes. IoT requires this kind of technology to allow secure communication among IoT nodes in heterogeneous environment. The BC in IoT may help to improve the communication security [39].

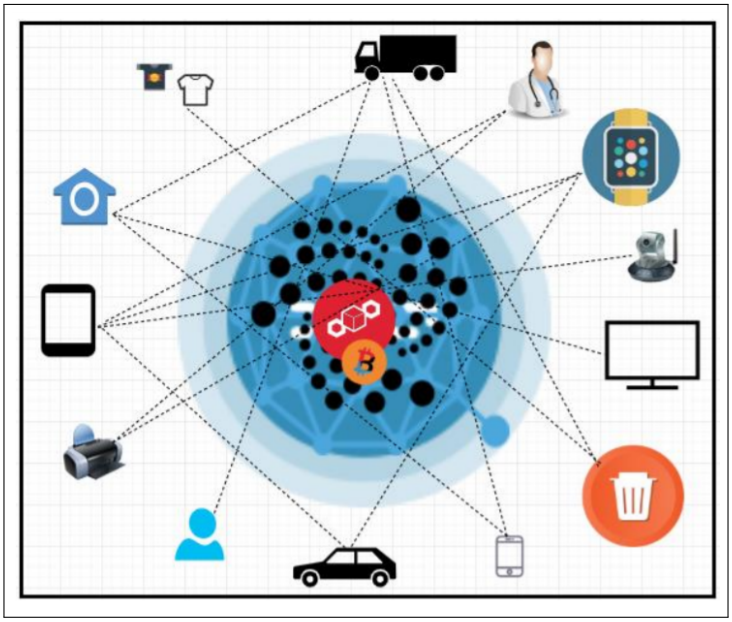


Figure 3.3: Blockchain relationship with the internet of things.

3.2.3 Application of blockchain in healthcare

Blockchain technology has gained significant interest in recent years with increased interest in many diverse areas, including the healthcare industry. Because Blockchain technology provides a secure and distributed database that can operate without central authority or administrator, this technology has also gained interest as a platform to improve the reliability and transparency of data. Healthcare through a multitude of use cases, from preserving permissions in Electronic Health Records (EHR) to simplifying claims handling [40].

CHAPTER 4

IMPLEMENTATION AND ANALYSIS

4.1 Problem

The most popular consensus mechanism, PoW, requires high energy resources to secure the chaining task. PoW makes sure that no one can take over the chain without achieving 51% of the total computational power in the network.

Other consensus mechanisms avoid high energy consumption by trying to secure the chain without using computational competition. The outcome is that energy consumption decreased.

4.2 Purpose

In this work, we propose a consensus protocol in an asynchronous environment. We control the energy consumption to ensure the synchronization by the application of several rounds of consensus, in each round the nodes make a Proof-Wait, i.e., show the PoW then make await.

4.3 Goal

The obtained results of this technique show a significant improvements by minimizing the energy consumption and making a progress state of the consensus. Fixing the number of processes, we got an energy gain rate equal to 21.62 % with five rounds and a gain rate equal to 25.41 % with ten rounds ,It is a fairly significant quantity.

4.4 Proposed protocol

As shown in figure 1, we divided the overall operation to reach consensus into several rounds.At the start, the node launches with the first round and looks for a solution for its own block, like the basic PoW algorithm. Once the solution is found, the node shares it and checks whether there are nine (9) other solutions found in the network for this round (for example, in a scenario of 10 solutions to find), if so, this node has the right to participate in the next round and restart the PoW. If not, i.e., there are not yet nine (9) solutions found by other nodes, this candidate node will wait until it receives the remaining nine (9) solutions. In the last round, the first node that will find the solution will be the miner, so that in the last round the protocol looks for a single solution.Initially, the ID Round is equal to 0. After that, the ID Round is equal to the sum of nonces found in the previous round. therefore, in each round there is a new ID so that the nodes will work on it.

Proposed technique could be defined by the following algorithm:

Algorithm 1 Consensus algorithm

Require: $NbrS$, $lastRound$

Ensure: $solution$

```
1:  procedure Cons(NbrS, lastRound)
2:      sols ← 0
3:      roundID ← 0
4:      nextRound ← 1
5:      q ← Queue()
6:      blockID ← random()
7:      while nextRound <= lastRound do
8:          while solution not found do
9:              search for solution
10:         end while
11:         sols ← sols + 1
12:         if nextRound is lastRound then
13:             print(I am the winner) , Exit()
14:         end if
15:         Broadcast(nextRound, solution)
16:         put on(q, solution)
17:         while sols < NbrS do
18:             Nothing
19:         end while
20:         sols ← 0
21:         nextRound ← nextRound + 1
22:         RoundID ← ∑_{i=1}^{NbrS} q(i)
23:         Broadcast(nextRound, RoundID)
24:     end while
25: end procedure
```

N.B : If two nodes succeed in finding the solution in the last round. In this case, the nodes must calculate the standard deviation of the solutions found in all the rounds for each of these two winners. The miner is the node which the smallest standard deviation.

There is another parameter that we can introduce here, which is the consensus state. For the moment, in which round do the processes work? Assuming that a process decides to leave competition if the other processes exceed it by a certain number of rounds, this will minimize the energy to be

consumed. Because he For a process, if the other competitors overtake him by two rounds at least, there is little hope of catching them. This process will lose energy unnecessarily if he continues the calculation In the rest of the rounds (Wasted effort = wasted energy).

4.5 Proposed protocol and demonstration

We assure four conditions of a consensus are: accord, integrity, validity and termination. In fact, we see that there is a fifth condition that must be satisfied in every consensus to be perfect. This condition is equality of opportunity (no domination). We will see in this paragraph that our protocol verifies these five conditions.

4.5.1 Accord

In the final round, in the event that there is only one winner, all the nodes will agree on him. In the case where there is more than one winner, for each winner, we will calculate the standard deviation of his solutions found during all the rounds and we will choose the minimum of these standard deviations. Mathematically, this value is unique.

4.5.2 Integrity

Once the Px process obtains a valid nonce (Blocki + nnonce1 < target), the process broadcasts it over the network with a specific date (timestamp). Of course, this nonce is concerned with a specific block, whether Bi+1, i.e. Px(Bi+1, n1). After having obtained a second nonce n2, the process will broadcast it (Px(Bi+1, n2)), we notice here that there are two nonces n1 and n2 for the same block Bi+1. In this case, there are several actions we can take, including taking the smallest nonce. Eventually, each process will participate with at most one value, so we have fulfilled the condition of integrity. Validity: For each solution received, the node will check its validity. The miner at the end is a network node, having a unique public address and resolving the PoW in all rounds.

4.5.3 Termination

Where PoW depends on the difficulty, its resolution is estimated at 10 minutes, and this 10 minute is more than enough for the propagation in the network and the candidate block will have been added to the blockchain. In the proposed technique, the difficulty is minimized and the number of rounds is increased, number of solutions. To ensure the termination in a finite time, we just need to balance between This factors (difficulty (Target), Rounds, Solutions). So, if the PoW has a termination, the proposed protocol also has a termination.

4.5.4 Equal opportunity

We have fulfilled all the conditions, but in reality, we have not reached an agreement, because the result is expected or known in advance (higher efficiency). Therefore, the condition of non-domination must be added for the agreement to be correct and just. only the few who had the most computing Power could solve the equation, so they still dominate mining. In the proposed protocol, we have reduced the difficulty. Therefore, those with limited computing abilities can find solutions in each rounds. this weakens the probability that the owner of the Power is always the dominant one.

4.6 Implimentation (execution in Python)

In our experiment, we simulated each node by a process, where we implemented multi-process programming. It is noteworthy that each process goes through one of the following four stages.

Figure 4.1: Computer en exaction

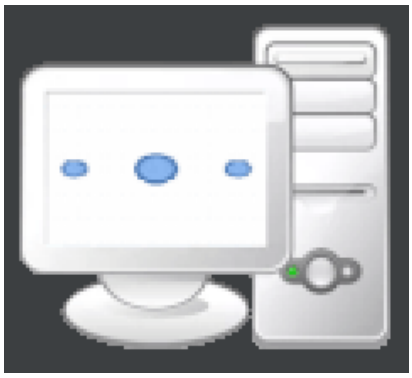


Figure 4.2: Computer en wait

Figure 4.3: Computer en fin

Figure 4.4: Computer winner

- During execution (4.1).
- During wait (4.2).
- The completion of the procedure (4.3).
- Is it a winner or not? (4.4).

In several tests, we created a different number of processes, each of them starts by creating a random number blockID random() (Line six in algorithm), this number considered as the ID of the own candidate block that the process is working on it. After that, each process follows the execution of the proposed algorithm **??**.

In the original PoW, each process will continue the calculation during the 10 minutes (until one of the processes finds the hash during this period). Contrary, in the proposed protocol there are two types of processes:

- Winning processes, are the processes that participate in all rounds.
- Processes that leave the competition after such rounds (for example, when the number of solutions for the the next round is satisfied).

In these two types, no process does the calculation during the 10 minutes, there is a proof-wait (first type) or a proof-abandon (second type). So, there are two main factors that must be handled, the number of rounds NbrR and the number of solutions to be found in each round NbrS.

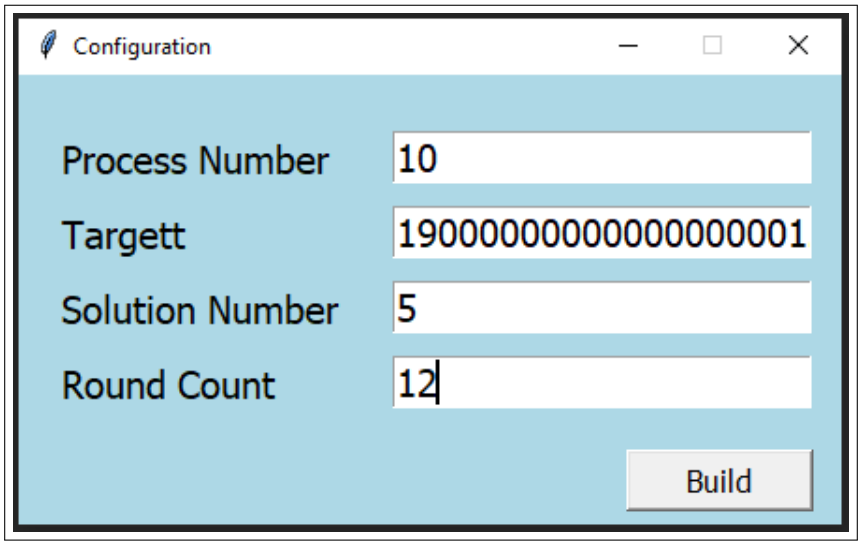


Figure 4.5: Configuration interface.

Whene using a twelve rounds and Ten processes, five solutions number (values not on appointment) 4.5, the test took about 5 minutes, (04:45:52 It's indicated that the target was 1.9E 74). we got the result in the Figure 4.7.

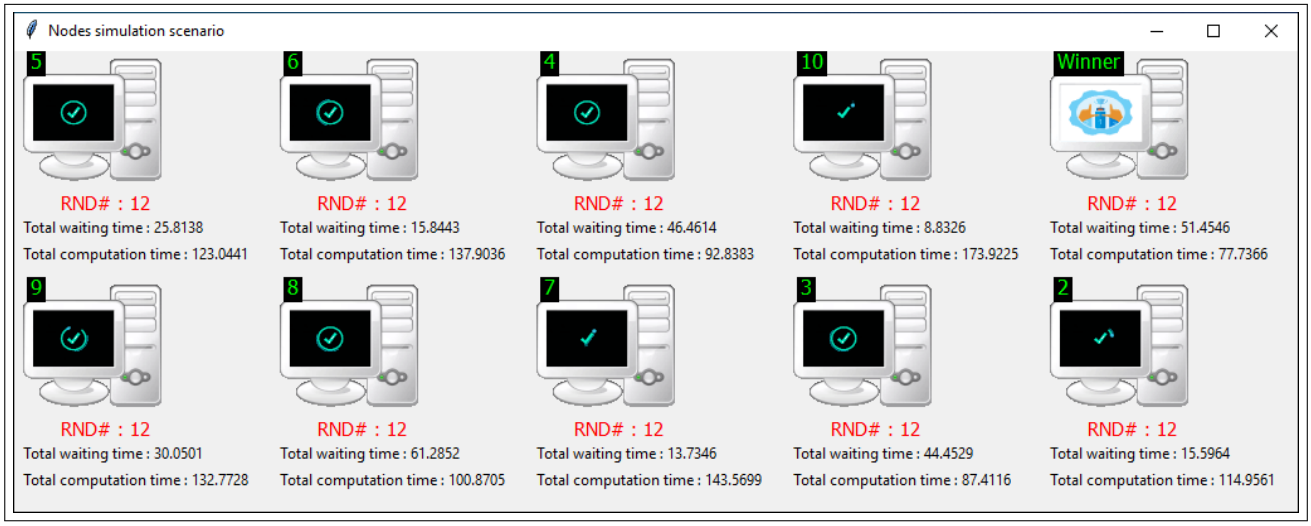


Figure 4.6: Nodes simulation scenario.

Then we transform the results obtained in Figure 4.7

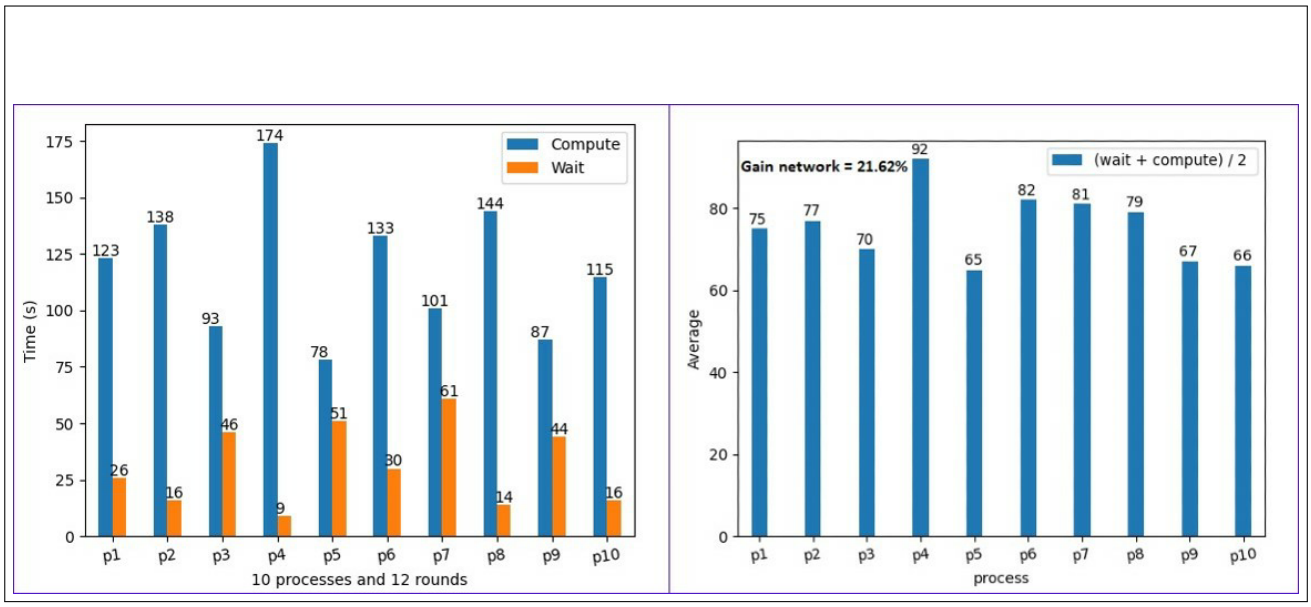


Figure 4.7: Compute and wait example execution.

4.7 Analysis

4.7.1 First scenario

In the scenario shown in figure 4.8 , we have set the number of rounds NbrR = 12, NbrP = 10, NbrS = 5 . We do put different values between NbrP and NbrS . We notice that there is really compute and wait in each round for most of the processes. Process 5 (5p) is the fastest because it has the lowest average sum time compute and wait (65) from the rest of processes as shown to the right of Figure 4.8 . For p5, the gain rate = wait / (wait+execution) = 51/(51+78) = 39.53%. On the other hand, process 4 (p4) is the heaviest process, because p4 having the highest average sum time compute and wait (92). For p4, the gain rate = 4.91%. We explain that last process who found the solution from each round it wakes up the others then it starts directly the next round with-out making any wait. So we can find a process that's not waiting (total time of wait zero), which means it's the last one of every round. In general, the average gain of the network is 21.62%.

4.7.2 Second scenario (NrbP)

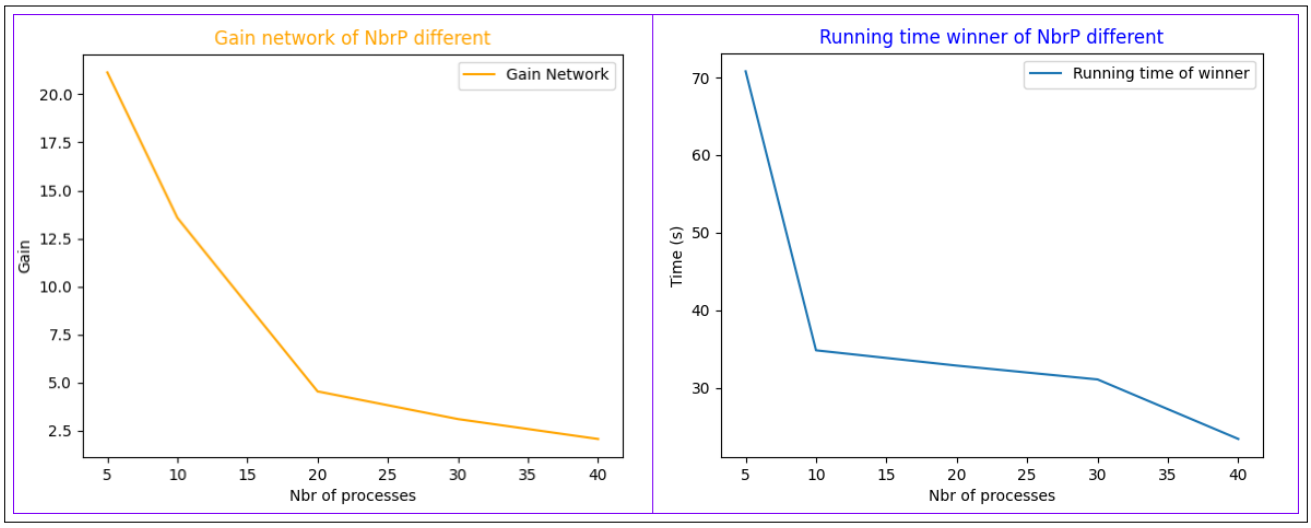


Figure 4.8: Network gain and running time of NbrP different.

In the second scenario shown by figure 4.9, we fixed the number of rounds (NbrR = 20), and number of solutions (NbrS = 3), we made several tests, in each one we increased the number of processes (NbrP = 5, 10, 20, 30 , 40) to study the effect of this increase and their influence on the gain. These tests have shown that the more many processes (number of rivals), Gain decreases exponentially. As well, for running time of winner (total wait + total compute) as shown to the right of the figure 4.9 .

The positive in this scenario is a decrease in running time to me winner, But on the other hand, The energy consumed is large (network gain poor).

$$\textbf{NbrP} \nearrow \quad \Rightarrow \textit{Running time of winner} \searrow, \quad \textit{Consumed energy} \nearrow$$

4.7.3 Third scenario (NbrR)

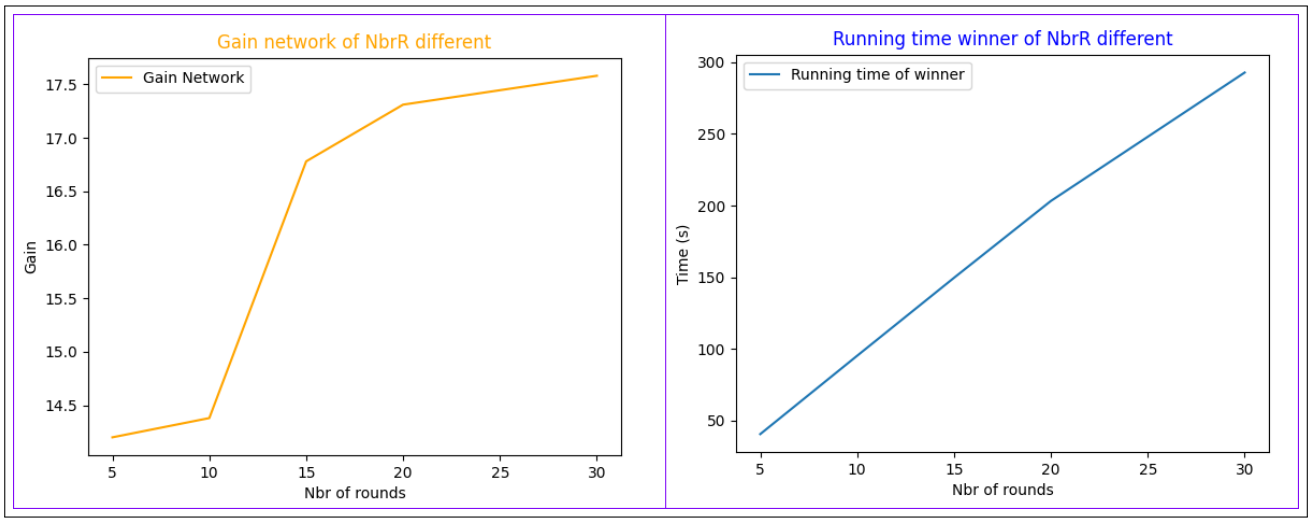


Figure 4.9: Network gain and running time of NbrR different.

In the scenario shown on Figure 4.10, we fixed the number of processes NbrP = 20, and number of solutions (NbrS = 5), we made several tests, in each of them we increased the number of rounds (NbrR = 5, 10, 15, 20, 30) to study the effect of this increase and their influence on the gain. These tests have shown that the more NbrR increases (rounds), the more the gain increases (Lower energy consumption rate) but not absolutely (14.5 -> 17.5). Rather, this increase converges to an upper bound related to the number of processes and then steadily after then. For example, in our test for processors 20, this upper bound was 17.58 (with 40 rounds 11.38 with 50 rounds and11.93, with 100 rounds 11.67) it fairly constant.

On the other hand, the more rounds, the more running time of winner increases absolutely as shown to the right of the figure 4.10 (from 40 to 293). This is due to the time total wait in each round, meaning that the process is at rest (sleep) i.e. more time to reach the end (solution) and less energy consumption.

$$\textbf{NbrR} \nearrow \Rightarrow \textit{Running time of winner} \nearrow, \quad \textit{Consumed energy} \searrow .$$

4.7.4 Fourth scenario (NbrS)

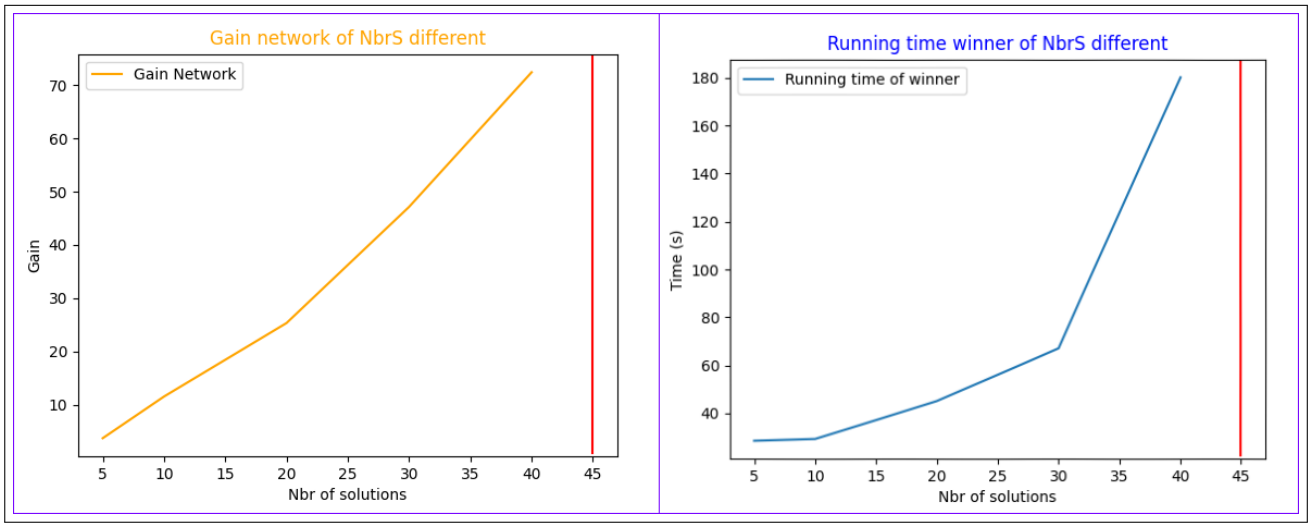


Figure 4.10: Network gain and running time of NbrS different.

In the fourth scenario shown Figure 4.11, we fixed the number of processes NbrP = 45, and number of rounds (NbrR = 20), we made several tests, in each of them we increased the number of solutions (NbrS = 5, 10, 20, 30, 40) to study the effect of this increase and their influence on the gain.

These tests have shown that the more NbrS increases (number solutions in the each round), Lower energy consumption rate absolutely i.e. significant increase in network gain (5 -> 73). Furthermore it, this increase converges to an upper bound related to the number of processes. Where if the value of the number of solutions equals the number of processes (NbrP = NbrS = 45), then the value of the network gain take the highest possible value (72.39). In the opposite side and to the right of Figure

4.11, we observe the more number solutions, the more running time of winner In a large percent Somewhat (from 28.52 to 180.23 With various experiences).

These two results (increase network gain and running time of winner) are interpreted as that each process affects the rest of the processes from where total wait Each of them and that in each round, for example if the number of processes is equal to number of solutions (NbrP = NbrS) when a process reaches P(x) to a round R(x) must each the waiting process until the condition is fulfilled (the rest of the processes arrive in the same round) and also for the rest of the rounds in this scenario, we note that the final waiting time for each process is very large (the first result) and thus more comfort for each process, which leads to less energy consumption (high network gain, Second result).

$$\textbf{NbrS} \text{ Lets Go To NbrP} \quad \Rightarrow \textit{Running time of winner} \nearrow, \quad \textit{Consumed energy} \searrow.$$

4.8 Summary of results learned from experiences

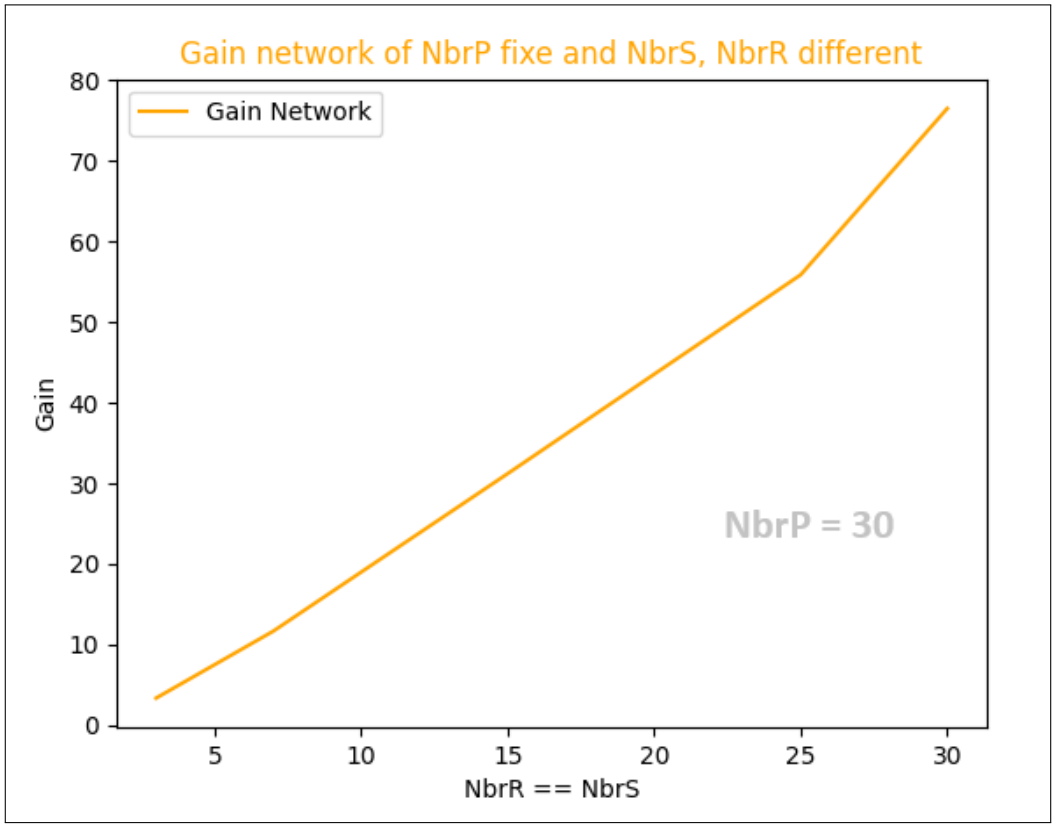


Figure 4.11: Relation NbrR with NbrS in gain rate.

In this experiment, two factors were manipulated, the number of rounds and the number solutions to be found in each round with fixed the number of processes NbrP = 30, this is to deduce the highest possible network gain, Figure 4.12 summarizes the influence of introducing these two factors (NbrR, NbrS) into a consensus algorithm. It should be noted here that when When the number of processes

is equal to the number of solutions and rounds NbrP = NbrR = NbrS = 30 the network gain is the highest possible value of 76.47. and No other set of values can achieve this (it can be, but it is a slight increase and is not calculated). For example, if the number of rounds is adjusted to the value 40, the value of the network gain is 78.15 and with 50 rounds it is 77.70, which are very close to the previous value (the maximum value) and conversely, value of solutions NbrS cannot be optimized because it takes the number of processes as its greatest value.

GENERAL CONCLUSION AND FUTURE WORK

In this work, we have proposed an effective and applicable consensus algorithm, whether it is especially in blockchain or in any setting where we need an agreement, it is adaptable to public or private access. We have studied its validity according to the four known conditions of the agreement, in addition, we have also added and demonstrate a fifth condition for the success of the consensus, which is equality of opportunity (no domination). We have shown the energy gain achieved by this protocol, Through previous results.We made several scenarios establishing in each one several tests, with the manipulation of two factors (number of rounds and number of solutions in each round). We studied separately the influence of each factor on the energy consumed. And also, the influence of introducing both factors comparing to the basic PoW.

In the future, The proposed consensus algorithm will be implemented on Raspberry devices for validation and showing its feasibility practically

BIBLIOGRAPHY

[1] proquest. Introduction. [Online]. Available: https://search.proquest.com/openview/9af6bebc02a48fbffe6712fb532edce0/1?pq-origsite=gscholar&cbl=32251

[2] author:imran Bashir. What is p2p. Accessed: 5-3-2021.

[3] docplayer. What is p2p. [Online]. Available: https://docplayer.org/203858551-Top-k-retrieval-in-peer-to-peer-networks.html

[4] ——. Disadvantages of pair to pair networks. [Online]. Available: https://www.codespot.org/introduction-to-peer-to-peer-network/

[5] sci hub. What is a distributed system. Accessed: 5-3-2021.

[6] medium. Distributed system theory. [Online]. Available: https://triaslab.medium.com/consensus-algorithms-and-fault-tolerance-in-distributed-systems-bf86d5aac45a

[7] boisestate. Types of distributed systems. http://cs.boisestate.edu/~amit/teaching/455/handouts/intro-handout.

[8] mbaknol. Characteristics of a distributed system. [Online]. Available: https://www.mbaknol.com/information-systems-management/characteristics-of-a-distributed-system/

[9] scribd. Advantages and disadvantages of distributed systems over centralized ones. [Online]. Available: https://www.scribd.com/document/99805534/Advantages-and-Disadvantages-of-Distributed-System-Over-Centralized-System

[10] tutorialspoint. Distributed system architecture. [Online]. Available: https://www.tutorialspoint.com/software_architecture_design/distributed_architecture.htm

[11] authors:Anand Ranganathan and R. H. Campbell. Introduction. Accessed: 10-3-2021.

[12] Overcoming complexity. Accessed: 20-3-2021.

[13] B. academy. The history of blockchain. [Online]. Available: https://academy.binance.com/en/articles/history-of-blockchain

[14] author:imran Bashir. Blockchain defined. Accessed: 1-3-2021.

[15] author:Iuon Chang Lin. The concept of blockchain. Accessed: 3-4-2021.

[16] data flair. dvantages and disadvantages of blockchain. [Online]. Available: https://data-flair.training/blogs/advantages-and-disadvantages-of-blockchain/

[17] javatpoint. principle of proof of wor. [Online]. Available: https://www.javatpoint.com/blockchain-proof-of-work

[18] bitconseil. dvantages and disadvantages of blockchain. [Online]. Available: https://bitconseil.fr/proof-of-work-definition-explication/

[19] geeksforgeeks. Main issues with the proof-of-work consensu. [Online]. Available: https://www.geeksforgeeks.org/proof-of-work-pow-consensus/

[20] sci hub. Introduction. [Online]. Available: https://sci-hub.do/https://link.springer.com/chapter/10.1007/978-3-319-67816-0_17

[21] coinmarketcap. What is proof of stake. [Online]. Available: https://coinmarketcap.com/alexandria/glossary/proof-of-stake-pos

[22] booksc. The principle of use proof of stake (pos). [Online]. Available: https://booksc.org/ireader/72727457

[23] investopedia. What is a proof of burn. [Online]. Available: https://www.investopedia.com/terms/p/proof-burn-cryptocurrency.asp

[24] T. R. C. Algorithm. What is raft consensus. [Online]. Available: https://raft.github.io/

[25] theblockchaincafe. What is proof-of-activity (poa). [Online]. Available: https://theblockchaincafe.com/what-is-proof-of-activity-poa/

[26] consensus. What is proof of elapsed time (poet). [Online]. Available: https://tokens-economy.gitbook.io/consensus/chain-based-trusted-computing-algorithms/poet

[27] burst coin. What is proof of capacity (poc). [Online]. Available: https://www.burst-coin.org/features/proof-of-capacity/

[28] smithandcrown. What is proof of importance (poi). [Online]. Available: https://smithandcrown.com/glossary/proof-of-importance/#:~:text=Proof$%$2Dof$%$2DImportance$%$20(PoI,on$%$20proof$%$2Dof$%$2Dstake

[29] investopedia. What is cryptocurrency. [Online]. Available: https://www.investopedia.com/tech/most-important-cryptocurrencies-other-than-bitcoin/

[30] intechopen. Advantages of the cryptocurrencies. [Online]. Available: https://www.intechopen.com/books/blockchain-and-cryptocurrencies/blockchain-and-digital-currency-in-the-world-of-finance

[31] A. P. D. R. HOUBEN and A. SNYERS. Mécanismes de consensus de la blockchain et leur relation avec les cryptocurrencies. Accessed: 14-2-2021.

[32] investopedia. What is bitcoin. [Online]. Available: https://www.investopedia.com/terms/b/bitcoin.asp

[33] B. S. Guide. How does bitcoin work. Accessed: 02-2-2021.

[34] startup.info. characteristics bitcoin. [Online]. Available: https://startup.info/characteristics-of-bitcoin/

[35] bernardmarr. The main differences between bitcoin and blockchain. [Online]. Available: https://www.bernardmarr.com/default.asp?contentID=1849#:~:text=Bitcoin$%$20is$%$20a$%$20cryptocurrency$%$2C$%$20while,while$%$20blockchain$%$20is$%$20about$%$20transparency.

[36] B. S. Guide. How are bitcoins stored and transactions made. Accessed: 13-4-2021.

[37] ——. Advantages of using bitcoin. Accessed: 01-4-2021.

[38] moneycrashers. Disadvantages of using bitcoin. [Online]. Available: https://www.moneycrashers.com/bitcoin-history-how-it-works-pros-cons/

[39] E. International Journal of Scientific Research in Computer Science and I. Technology. Blockchain and its role in (iot). Accessed: 10-5-2021.

[40] sci hub. The use of blockchain in healthcare. [Online]. Available: https://sci-hub.do/https://www.ahajournals.org/doi/full/10.1161/circoutcomes.117.003800