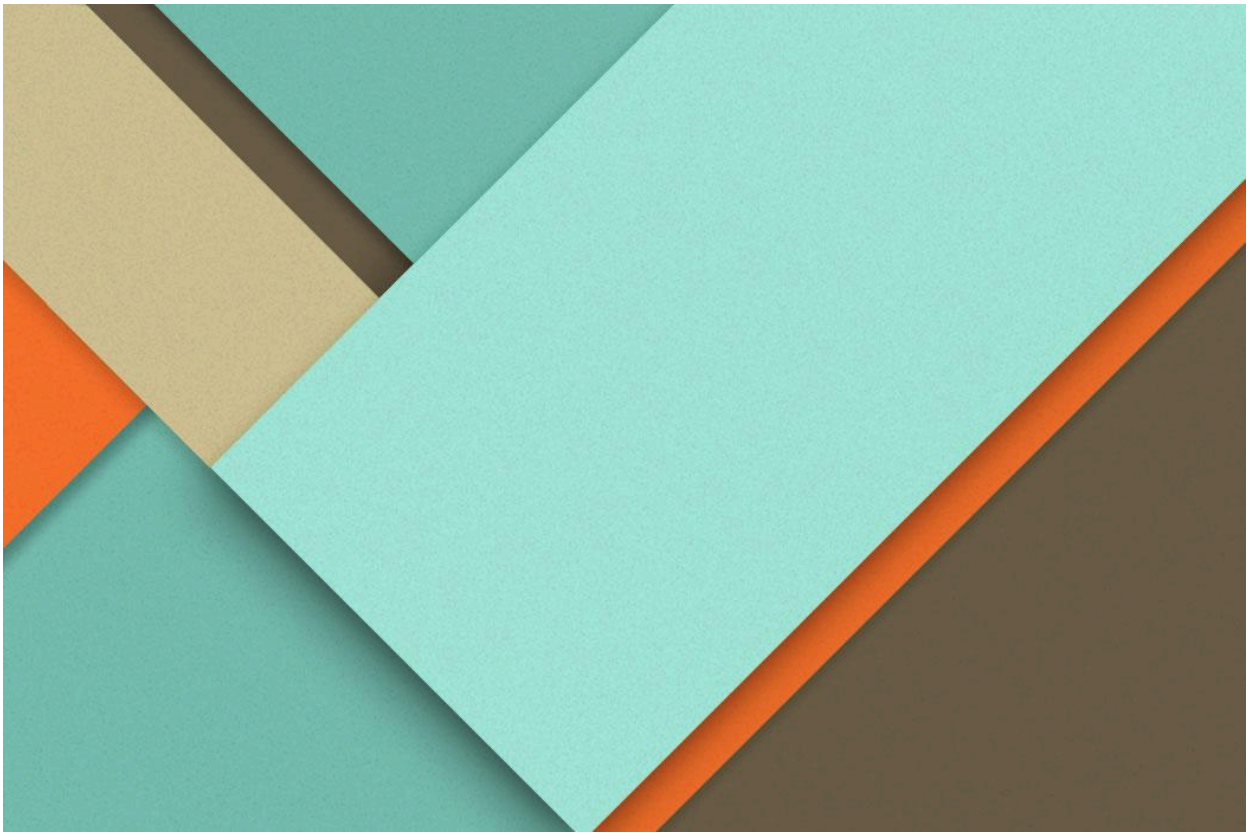


Peer-to-peer Network in Massive Multiplayer Online Games

Haoyi Pan(Kouki), Anthonese Mitchell, Kaimin Wang

Abstract

Massive Multiplayer Online Games (MMOGs) are becoming increasingly popular in recent years. They are becoming a trending option for entertainment. With larger and larger numbers of concurrent users joining the games, the current network architecture can not always satisfy the requirement of MMOGs. Moreover, MMOGs operators' profit is obtained from the fees users pay to access the various game contents. Thereofor, offering an acceptable level of service at low upfront cost while ensuring the game's network architecture able to sustain a large number of users is upcoming development direction for MMOG operators. In the meantime, the last decade has seen the rise of P2P technology targeting large-scale multiplayer online games since P2P-based architectures are good at quickly connecting associated peers among massive user platforms. In this paper, firstly, we will go through the history of P2P and MMOGs and present the common used network architectures with their characteristics by comparison in modern MMOG. Then, we will focus on the demonstration of several P2P-based techniques and protocols that can be applied on MMOG to improve the game performance with detailed analysis in order to show the advantages and feasibility to play MMOG over P2P architectures.

Table of Contents

Introduction

Nowadays, online gaming entertainment has achieved enormous success in terms of ecmonical return and popularity among the society. Due to the economic growth of the field, online games have acquired a tremendous amount of attention, in particular regarding Massively Multiplayer Online games. Today's architectures for MMOG mainly rely on a client-server model. This centralized approach is simple to create and easy to manage from MMOG operators' perspective. However, a whole game relying on one server will eventually cause critical issues such as hitting the scalability limits and depleting the data resources of the central server. Therefore, researchers will naturally start to lay their eyes upon the P2P paradigm, which has already reached success in downloading, video chatting and streaming fields. In the last decade, several departments have already started to implement P2P-based network architectures for MMOGs. We have to admit that there are a huge array of problems facing up against applying P2P into MMOG's network architectures. We will discuss the association between MMOGs and P2P architectures through history records, comparison of common existing architectures, and some research projects of P2P-based techniques and protocol contributed towards either integration of MMOGs and P2P architectures, or P2P-based algorithm in terms of helping to improve MMOGs' system management and network tolerance latency.

History of P2P Network

ARPANET

ARPANET (Advance Research Project Agency Network) was developed in The United States by the Department of Defense (DoD). ARPANET consisted of multiple hosts and each was deemed essential. The first version of

Telnet was stemmed from ARPANET. Doom was one of the first networked games that utilized USENET and FTP sites. Many of the protocols used such as FTP, DNS, HTTP and SMTP are based on the assumption that hosts connect directly with each other to share data and rely on the network to ensure it happens. Although the simplicity of its routing system was effective, ARPANET was not able to keep up with advancing technology.

Napster

Napster, developed in 1999 is well-known as the pioneer of P2P networking. It used a centralized index that allowed users to search and store files on a network. Peers are associated with a central database where information about the content they want to share is published. A user seeking specific content queries the database, retrieves the IP address of the nodes where the data is located and downloads it directly from one of the nodes. This centralized approach made it simple to identify files and keep track of those uploading and downloading files.

Gnutella

Gnutella, an unstructured overlay network, started the second-generation peer-to-peer networking. This model has been adopted by others since there was none like it replacing centralized systems with decentralized systems. Typically, unstructured overlays are differentiated by how search requests are propagated and by variations in the creation of links with nearby peers. Gnutella also implemented the flooding mechanism where queries are forwarded to all linked peer nodes to verify their request. Additionally, SOLIPSIS was a massive multiplayer virtual world created in the early 2000s that utilized Gnutella. It optimized this topology by gathering entities which are close to the game-world on the P2P network. The distance that messages have to move on the underlying P2P network is therefore decreased. However, Gnutella proved not to be scalable.

BitTorrent

BitTorrent is the third generation of peer-to-peer networking implemented. The premise of BitTorrent was to allow nodes to simultaneously download and upload files. Even though BitTorrent operates as a decentralized system, it still relies on a central server which is called a *tracker* for coordination of users and to keep track of files' location and availability. The gaming developer and publisher Blizzard Entertainment Inc. used peer-to-peer for its multiplayer setup when Diablo was initially released. A player would serve as the host and the other as a client. Another service offered by Blizzard that also used peer-to-peer for gaming was Battle.net which enabled a voluminous number of concurrent players with a server for chat and another for matching players.

Commonly Used networking architectures in MMOGs

Introduction of Massively Multiplayer Games and Networking Architectures

With the development of networking architecture and the prosperity of the video game industry, Massive Multiplayer Online Games (MMOGs) are becoming increasingly popular in recent years and have shown the dominance of the current video game markets. It is well-known for allowing thousands of players to share a single game world. MMOGs consist of role-playing games (RPG) or real-time strategy (RTS)/RPG hybrids. The most popular examples are World of Warcraft, EVE Online, and The Elder Scrolls Online. These games' companies establish multiple servers or single servers in different regions across the world or the whole world so that people can meet and play together in various virtual worlds with strangers or friends.

The basic structure of most MMOGs are very similar to each other. Players firstly created a character assuming a role in the virtual game world. The world settings are getting more and more diverse nowadays; it could be in western medieval period, modern present warfare, or future cyberpunk styles. The player will experience the game world through a game avatar, which is the representation of their character in the game. Then, players can control the game character taking on missions either alone or as part of a group that requires them to enter different regions

of the game world, interacting with different players, and earning rewards by defecting enemies or finding, gathering requested objects. During this process, characters will obtain experience and items for leveling up and enhancing gears to get stronger, and then they can fight more powerful creatures in the game or participate in PVP contents.

The rapidly growing MMOG market attracts countless online games appearing, but the networking architectures implemented in those games are basically the same. Three main architectures that are typically used in MMOGs are: 1) client-server architecture; 2) multi-server architecture; 3) Peer-to-peer(P2P) architecture. Every architecture has their unique potential to deal with different concerns existing in MMOGs. In general, we analyze typical characteristics of MMOGs, and review and compare current common networking architecture in MMOGs.

General Game States

For a typical MMO game world, it is made up of immutable landscape information (the terrain), mutable objects such as weapons, armors, food and tools, mutable landscape information (breakable structure: pillars, walls, windows), characters controlled by players (PCs), and non-player characters (NPCs) who behave depending on preset automated algorithms. NPCs can be neutral, or either rallies or enemies. They are the main components in the game world to help players feel immersed in the gameplay and progress the game stages.

The terrain is the foundation of the game world map. It consists of all immutable elements in the game which players are unable to interact with. However, they are usually designed as real physical objects in the game that can be used as boundaries to separate the world into several regions and limit the movement areas for the players.

The state of a player can be categorised as positions and condition of avatar. The game system needs to always update players' position so that they can load corresponding resources in specific areas that players currently are in. Avatar states are referring to players' characters' health, abilities and inventory's items. This information is often persistent and carried along the player's account from one login session to another. The NPCs or game objects can also have avatar states that may stay persistent or temporary depending on the purposes of the states. Such as when

fighting a monster, the player's character can apply a temporary negative effect on it or a friendly NPC has a favorable impression meter towards the player that will stay persistent even if the player changes to a different character.

Generally speaking, a player can take three actions in the game world: interaction between players and objects, interaction between players and players, and position change of player's character. Players interacting with objects or other players would always lead to avatar state changes: A player enchants a weapon with other materials and succeeds. This action will change the weapon status and subtract the amount of materials and improved weapons may provide new effects on characters. For players vs players mode, one player can damage another player in order to drop the health to zero to win the match. The health and abilities parameters would rapidly change during this process for both players.

In modern MMOGs mechanics, some other game states are periodically refreshed. Grinding taks up a big part of most MMOGs, which means there would be millions of mobs, boss, or wild animals being killed by players to gain experience, items or in-game cash. This is usually implemented as a daily or weekly reset or as a periodic respawn of "consumed" collectable items and hostile NPCs.

Some MMOGs that use a centralized server model would be further divided into small servers in one region to avoid game servers being over-populated. Players are allowed to switch from one server to another but this process requires extra mechanism to support. Except for avoiding overcrowded issues, this would also control over the data so that clients can easily load and run the game with smaller memory consumption to their devices.

The effect of network latency on player gaming experience

Network latency, normally described as lag, is the most important aspect that players are concerned about when they play a MMOG. Player tolerance for network latency varies based on the genre of the game. In general, a game where players use point and click as the main approach to lead one character or a group of units has higher latency tolerance than action RPGs which have a set of control keys to perform diverse combos of action sequeues.

First person shooter games such as Overwatch or action MMORPG such as Black Desert where players directly control the characters' movement and attack can only tolerate latencies less than 180 milliseconds. Any longer lagging would cause inconsistent gameplay for players and even lead to losing the game. Otherwise, real time strategy games like Age of Empires Online can tolerate up to a few seconds of network latency based on the focus of the game types: it is focused on strategy rather than fluid control performance. The traditional turn-based RPG would also have a high tolerance latency since it only needs to give the order to the character and the algorithm helps do the rest. With the development of electrical sports and network communication techniques, the network latency has been mitigated gradually to offer and guarantee a better gaming experience for professionals and casual players.

Currently Commonly used Networking Architectures

1. Client-server architecture

Today's architectures for MMOGs mainly rely on the client-server model. In this model, players connect to a centralized server using their own client software(PC, Console, Switch). The server works as a central controller: it contains all the copies of avatars and objects to share with clients, and it maintains the global game states. This centralized approach enables a straightforward management of the main functionalities of the virtual game world, such as user identification, state management, synchronization among players and billing. This typical client-server model offers a simple way to deploy, maintain and modify the game world since the central server is designed to be able to control everything.

The characteristic feature of client-server model:

1) Consistency

   Since all users need to connect to the main server in order to play the game, this would avoid inconsistent scenarios that could usually happen in a distributed environment setting, in which the same actions may be taken at different nodes and conflicts appear due to network latency or others. All users' actions are

recorded and managed by the central controller in client-server architecture and the concurrent and possible conflicting operations will be executed in correct order so that players notify only the first successful action. This guarantees all the critical moments of the game will proceed in the right way for the sake of fairness.

2) Simplicity

MMOG architecture requires to be as simple as possible for programmers and operators to consider handling massive players to play their games concurrently. From the programmers' perspective, they prefer to build simple and well-constructed architecture so that they can easily code and update with the consistent coding architecture. From the perspective of operators, their profit is linked to manage and maintain the system of the game while thousands of players getting involved. Therefore, they would like to operate the simpler architecture to maximize the ratio of efforts and payroll.

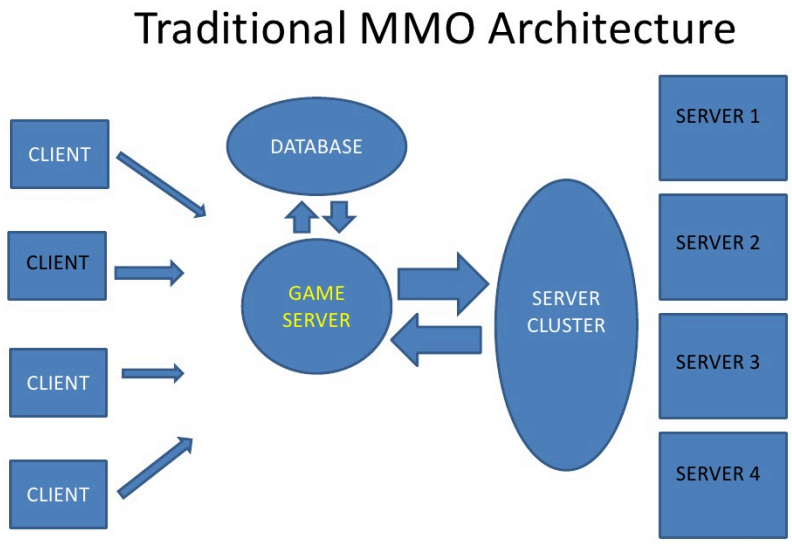
3) Ease for development, maintenance, and modification

MMOGs usually take two or three time longer of development period than developing a single-player game. Thus a fast and easy development method should be taken into consideration. Customer support and maintenance are the most extremely important part to promise a client-server architecture keeping alive. The central server provides an easy structure for support service to retain them at low cost and for building easy algorithms of programs to de\tect malicious behaviours of the player's states to be fair for all players. It is more convenient to implement these with a client-server model. MMOGs are also subject to constant modification in regard to updates, bug fixing and many more. With a high level of consistency and unified system monitor protocol. This will make every modification easy to add and track, and also avoid hacking.

Overall, when consistency control and simplicity are the main concerns, client-server architecture is the best option

to pick. Majority of MMOGs are based on client-server architecture due to its simplicity. It is way simpler to write a centralized architecture than peer-to peer architecture. The main server helps take care of everything; it is the highest level of control over the game world. Developers can easily change and update the game state and have control over the necessary updates. The management and monitor would be easy to observe by operating the central controller. The consistency is highly guaranteed in this architecture compared to the rest of networking models.

However, with larger and larger numbers of concurrent players, centralized architectures are easily hitting their scalability limits. One server's resources can be quickly used up. This will make the maintenance and modification cost more and more in terms of economical return for game developers.



Most wide-used Multi-server Architecture for MMOGs
Image Source: https://pt.slideshare.net/ACMBangalore/using-the-cloud-to-build-a-mmorpg/2?smtNoRedir=1

2. Multi-server Architecture

Multi-server architectures expand the common client-server architectures. It is also known as cluster-based centralized architectures that are applied with virtual data locality technology to increase scalability of MMOG.

Typically, this architecture divides the global game world into several regions, and each region world as one client-server architecture. It means that every server is presonsidlb for its own set of clients and has its own game world that is not interfered with by other servers. The most common approaches are zoning, mirroring and instancing. Players can communicate with other regions over zoning and mirroring, and mirroring offers a region that can be replicated in multiple servers at the same time. Instancing makes each replica a region independent. Therefore, players are not allowed to communicate among different replicas in the instancing mechanism.

1) Scalability

   Scalability is the major concern of MMOGs. Multi-server architectures use cluster-based techniques to split the large virtual world into smaller regions. This accommodates the growth of the users as the popularity of the game increases to allow for dealing with peak numbers of concurrent users while providing the ability for further expansion of the game capacity.

2) Fault Tolerance

   MMOGs users expect the game servers to be stable and constantly available with minimal downtimes. If a game central server goes down, it will cost over ten thousands dollars per hour from the lost. Multi-server structures always have a backup plan to deal with outages on the particular down server to avoid total blackout of the whole game.

3) Consistency

   Multi-server architecture is inherited from the common client-server model, which can guarantee consistency control over one main central server. Multi-server can guarantee the set of clients for each region server sharing the same updates and contents. However, the consistency between servers in each region may suffer from differences, but this generally will not affect the global game states when players play in their own region server.

Typical multi-server architecture will use event ID tracking mechanism to tackle the consistency requirements. A

simple, easy to implement scalability algorithm is commonly used that addresses the growth of the system and adds new nodes to increase the size of the virtual world. Mirroring scheme is used to cope with server failure so that replicas can be replaced as the backup. In general, multi-server architectures are the mainstream pattern that most MMOGs follow nowadays. This architecture covers all major issues that a MMOG can raise in development. The drawback of it is that the problem of resource provisioning is still not addressed, as communication bandwidth from all users is run through at one data center. This requires a static provisioning of a large bandwidth capability, and therefore leads to over provisioning issues in which MMOG always needs to maintain the resources capacity even when the load of concurrent users is not at its peak.

3. Peer-to-Peer Architecture

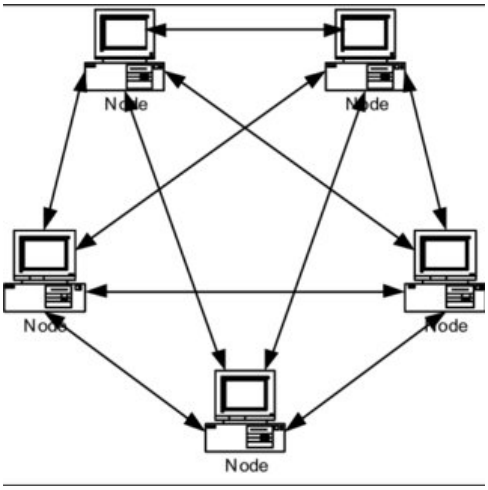


Image source: https://www.researchgate.net/publication/220686424_Agents-based_modeling_for_a_peer-to-peer_MMOG_architecture

Peer-to-Peer (P2P) architecture is proposed to improve scalability. Each peer world has both client and server, and it will handle some parts of the copies of avatars and objects, as well as maintain some parts of the global game state. By reducing the load on centralized servers, P2P-based architecture presents its own unique advantages.

1) Scalability

   As mentioned at the beginning, P2P techniques are inherently scalable as the available resources grow with the number of users since they are the servers as well.

2) Self-management

   When a peer fails, P2P has the ability to self-repair and reorganize like the multi-server characteristic to avoid the outage of the whole network; hence, the stability and robustness is greatly strengthened.

3) Prevention of Traffic Congestion

   The network traffic is distributed among the users involved, making it difficult to cause transmission getting stuck.

4) Operation Cost

   Moreover, peers are the servers themselves that help manage the load of the whole network and game operators have no need to add expensive servers to maintain the game states. This will add no extra cost to the game developers for maintenance fee.

For MMOG operators to control and monitor the game over a central server, this is considered as too costly to deploy clients based on their points of interest. In P2P architecture, clients serve as a node inside the network flow and client nodes are allowed to communicate directly with each other and perform part of the work as a server. In the case of MMOG companies, this way can greatly reduce the cost for the operators to manage and maintain the networking system since they also need to raise up the bandwidth and even power consumption as a static data container in order to cope with peak performance of concurrent users. With the fact in MMOG that the users can be located in anywhere across the world, and the number of users is often undetermined and changed dynamically, P2P architecture is more scalable solution since client nodes are more relying on each other, rather than relying on the central node in the client-server architecture. However, the lack of a central authority hinders security and anti-cheating enforcement in the game. In addition to security concerns, user machine environments typically have various, different constraints on computational, storage and communication capabilities. This makes the architecture require extra mechanisms to exploit their difference in order to group an effective peer cluster for better performance, otherwise, peer failures and disconnection could happen all the time.

4. Other Architectures

There exists other architectures to aim at fixing specific issues in MMOGs. In recent years, there are many research papers proposing to combine P2P technology with a client-server architecture, which is considered as a hybrid architecture. The decentralized distributed systems are also the hot topic to apply into different networking industries, such as architectures like Service-Oriented Architectures (SOA) can be used for MMOGs to fulfill the high bandwidth concern with a specific middleware based on open source DSS to make all functionalities of the global game states in MMOG running at real time.

Most hybrid or other technique architectures are usually still in the development or research phase. They are not qualified at the moment to be widely used in MMOGs and still need an amount of extra new mechanisms for the integration.

Proposed P2P approach for MMOGs

As mentioned, multiplayer computer games often rely on client server systems, but it has limited scalability and if the server goes down, all the players cannot play the game. Also, replicating servers creates some issues like synchronising information among servers. Therefore, several P2P approaches have been proposed to solve these problems of client server model, and we will introduce 3 of them as examples.

Central Connection(CC) approach

This model is a hybrid of P2P and client server architecture, and this solves three major problems of pure P2P solution in MMOGs[1].

- The network could be divided into smaller networks, and it causes game data loss and state.
- There is no main connection point for new coming players, so it has no stable way of connection.
- The game has no central authority, which leads to cheating in the game.

Therefore, Central connection(CC) is designed for avoiding network splitting, reducing cheating and maintaining a stable connection. This approach is a group of two algorithms

- CC assigning algorithms
- Cheating reduction algorithms.

Table 1: Player tracking table

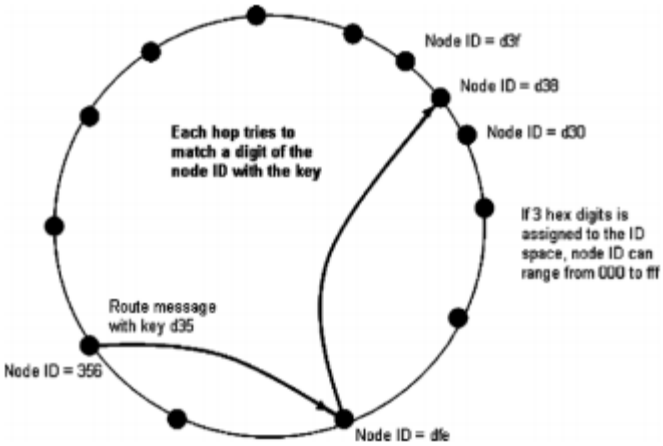
Player ID	IP Address	RAM (in GB)	GPU Speed	GRAM (in GB)	CC RANK
Player A	192.168.0.0	8	1.3 GHz	1.5	0
Player B	192.168.0.4	16	2.7GHz	2	0
Player C	192.168.0.2	6	2.3GHz	2	0
Player D	192.168.0.3	4	1.8GHz	1	0

In CC assigning algorithms, the system creates a tracking table of peers in a session like Table 1[1]. This table maintains peer's game information and device detail to decide CC rank which is used to pick the highest performance PC as a CC to the session. Then CC will act like a leader peer in the session, so one CC should always exist in a session. When the current CC is about to leave the session, the next highest CC rank PC will be a next CC. In the cheating reduction algorithm, it is possible that CC monitors every peer's action, but it will go against the fairness principle of the P2P system. So instead, the system pick P highest CC rank peers as panel, and P can be calculated by $P = I * N/100$where I is the percentage of panels required in the session and N is the total number of peers in a session. Then, for each cheat proning actions like getting items or exp, panels will vote to CC if the action is valid or not. Finally, CC will decide if the player's action is cheating or not using the votes from the panel. These two algorithms complete the CC approach.

Pastry and Scribe tree approach

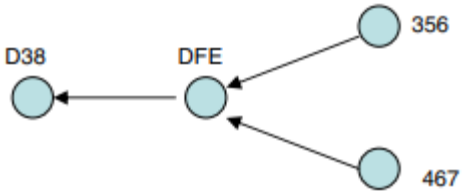
In this approach, researchers developed a peer to peer support scenario based on Pastry considering actual MMORPG network patterns[2]. To achieve smooth routing in a scalable environment, Pastry like Figure 1 is used for this approach.

Figure 1 : Pastry overlay



Due to asymmetric allocation of upstream and downstream bandwidth, they use Scribe which is an application level multicast tree. First, one node is selected as root, and when more than 2 nodes try to reach the root using the same path, the multitree is established.

Figure 2 : A scribe tree example

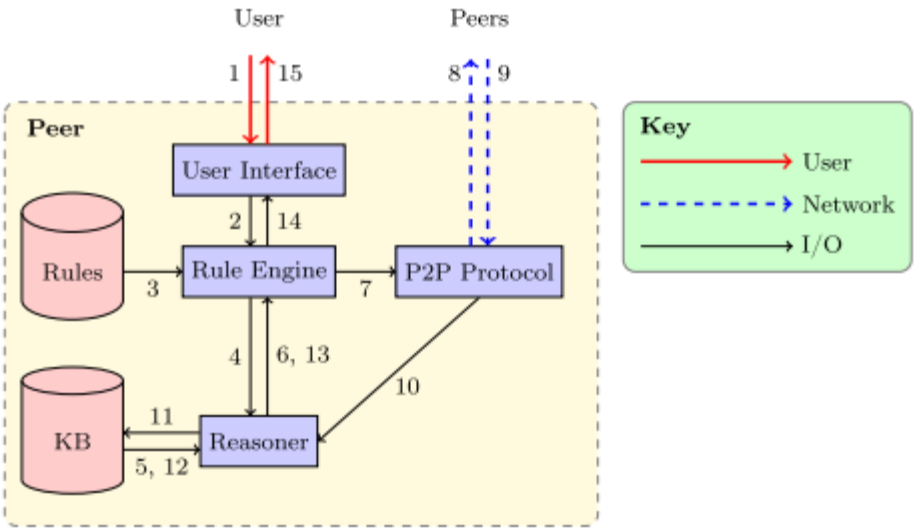


One example is given in Figure 2. D38 is root and 356 and 467 are the peer nodes. This tree was formed when 356 and 467 tried to reach D38 through the DFE. In regular client-server connection, the server generates 10 times larger traffic than the client. For example, in the 3D MMO game Lineage2, the server's upstream requirement is 140Mbp while downstream is only 9Mbp[2]. So in the simulation, they use Scribe multicast to solve asymmetry problems in P2P. The results of the simulation have shown that the Scribe tree performs well in bandwidth stress

among peers. Also, the scalability of the tree was good since even if the number of peers increased 5 times, the level of the tree only increased by 2[2].

Knowledge based approach

Figure 3 : Peer architecture



Last proposed approach is a knowledge based approach on P2P games[3]. Unlike client based MMO, each peer has to have storage for containing information about the current state of the game. Each component has the following characteristics.

- User Interface: Responsible for the interface between the user and the game; accepting user commands to be used as actions and providing feedback about the game state for the user to make future decisions on actions.
- Rules: Holds the game rules which dictate how the game is played and how the game progresses depending on the inputs given.
- Rule Engine: Processes rules as well as attempts to run actions.
- KB: The KB (knowledge base) represents knowledge about the state of the game as well as knowledge about the network.

- Reasoner: Provides knowledge to the rule engine so that it is able to process which rules apply. The application of rules generates fresh knowledge which is passed back to the reasoner so that it can update the knowledge base.
- P2P Protocol: Handles any messages to be sent to and received from the P2P network and to handle any information to be used by the peer or to be passed on to other peers in the network.

Figure 3 shows how components will interact with each other.

P2P PROTOCOLS

The goal here is to examine existing P2P protocols that would be compatible with MMOGs and compare them to determine their feasibility for market implementations and efficacy in mitigating bottlenecks. According to study, latency and game state management are the key problems that need to be addressed in P2P gaming [4].

LATENCY

Delays must be held at or below the threshold of human perception. The Distributed Interactive Simulation (DIS) and the High-Level Architecture (HLA) states that the delay should be around 200ms [5]. Delays are exacerbated by these two reasons: network and synchronization delay [6].

Network delay is defined as the time it takes to move and process data across the network. This latency occurs in a client/server model because of the server's lack of CPU power or bandwidth [6]. This decreases the amount of hops needed for players to communicate and eliminates the possibility of the server being a bottleneck.

Synchronization delay is defined as the time it takes to gather all players' activities and synchronize them [6]. When it comes to these types of delays, a play out buffer can be used to ensure that all players have the exact delay. Another consideration is an MMOG's total bandwidth use. It is imperative to consider the necessary bandwidth when reviewing various network structure approaches.

GAME STATE MANAGEMENT

Games are described by a series of states, which are changed by events as the game progresses. The system of a P2P game must ensure the game state is consistent for all participants.

An overlay is built to control the states in P2P gaming. The topology of it has a significant impact on how states are handled. Static and dynamic are two methods of overlays that should be considered. Peers are arranged in a hierarchical way in a static overlay that is independent of the player's interest. The overlay routes events and state accesses using a distributed hash table (DHT) [7]. For dynamic overlays, peers create the overlay based on their shared preferences and player experiences which can shift over time.

There are two types of connections that can be used, either unicast or broadcast. With unicast, it has low latency between linked peers, but they often demand more bandwidth. Multicast methods will help reduce bandwidth consumption, but this is contingent on the structure used to determine which peers the messages will be transmitted to [4].

TYPES OF P2P PROTOCOLS

N-TREE

The N-TREE protocol main concentration is on event propagation and handles peers based on their area of interest across the network. It is a tree data structure with N dimensions of every inner node and nodes can be either parent which are just called nodes or leaves which are nodes with no children [8]. When players interact with the game, it causes a trigger of events [8]. The N-Tree broadcasts the events to all peers that are within the affected region. In the N-Tree, peers are found using a distributed hash table. When join messages connect with a peer within its domain, the new peer enters the tree at that location [8]. Once a player logs off and exits the network, a leave message is delivered to all leaves in that player's group. A new leader will be selected if that peer was the original leader. When a new leader is being installed, the parent can queue events to retain state management, but this will create latency.

This protocol would result in players engaging with other players near to them first to decrease latency. As a result, synchronization between them and their peers are maintained.

QuON

QuON [9] is a protocol that, like N-Tree, is built on a quad tree and is designed with the player's actions in mind. Every player creates a quad-tree out of his own and all their identified neighbors' positions. There are three types of neighbors: direct, binding, and soft-state neighbors. Direct neighbors are players whose position is in the player's area of interest. Binding neighbors are tasked with relaying critical state knowledge around the world. Soft-state neighbors are used to ensure that all players who need to communicate over a greater distance than the field of concern will do so. Based on their location, every group ensures that their game states are synchronized. The quad tree and neighbor classifications can be remade if a player passes or receives a movement change but since trees are localized, it will not result in network overhead. The issue with QuON network structure is that network loss is not taken into account. To keep the network going, a peer can keep a list of old binding neighbors as backups in case the new binding neighbors are gone.

VAST

The VAST protocol is found on the Voronoi diagram [10]. The world is designed by creating a Voronoi diagram that is divided into tessellating regions with one player in each. Neighbors can be categorized using this type of diagram in a manner comparable to the QuON structure.

A player has a local copy of the diagram that they can keep up to date. Unicast links will be established between both neighbors, implying that nearby players will have a better chance of

being synchronized and reduced latency.

When a player wishes to enter the network, a join message is sent to a random peer who then relays the message to a peer whose location is closest to the joining player. This is done until the peer hosting the area in which the new player lives is located. It is imperative to note that this will all take place on every player's local computer with no further contact required than a simple move order. To exit the network, the peer must alert all its neighbors. The ones that receive the message will delete that neighbor from their lists and refresh their Voronoi diagram. The P2P overlay is continuously retained because the Voronoi diagram is recalculated.

PERFORMANCE AND ANALYSIS

In this paper [11], the writers ran tests to see how the P2P protocols can handle the various world sizes,the simulations were conducted with a fixed target node count of 512 players. Players also enter and leave the network as the simulation progresses.

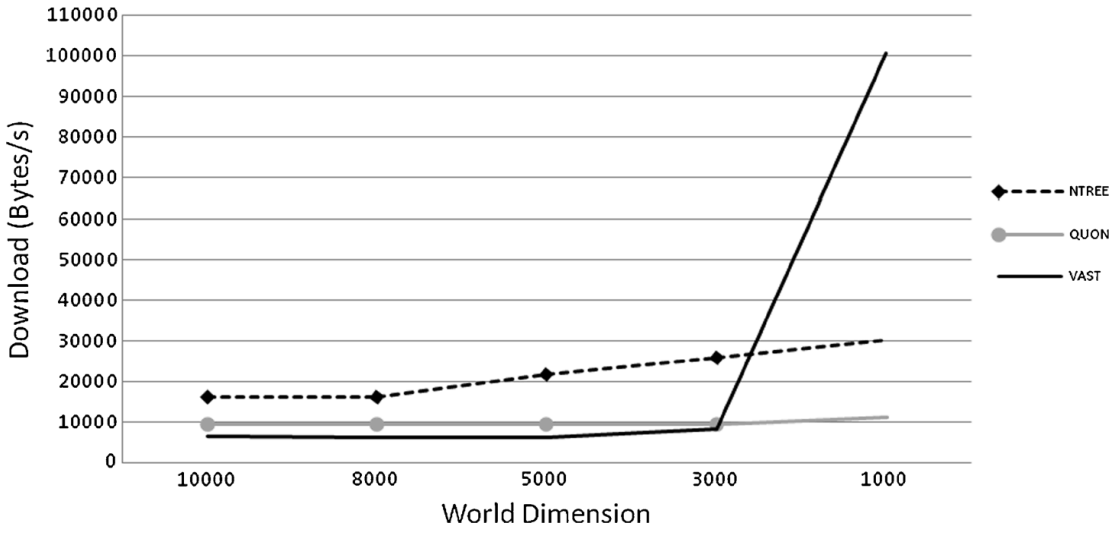


FIG. 1

Figure 1[11], depicts the average rate of download used by the node to collect all messages needed to sustain itself for various dimension settings. With players scattered around a wide area, 10,000 by 10,000, VAST requires the least amount of bandwidth to download messages. The population density rises to 0.512 while the world dimensions are as small as 1,000 by 1,000. The additional messages that require transmission in this densely populated area have a significant impact on the download rate for VAST making it unable to scale above a certain density range. QuON succeeds in terms of scalability. The bandwidth consumption of N-Tree is relatively low, but the steady increase indicates problems in crowded game areas. Figure 2 [11] below, depicts the uploaded bytes per second in simulations. Since everything sent by one node is received by another, the download and upload bandwidth results mirror one another.

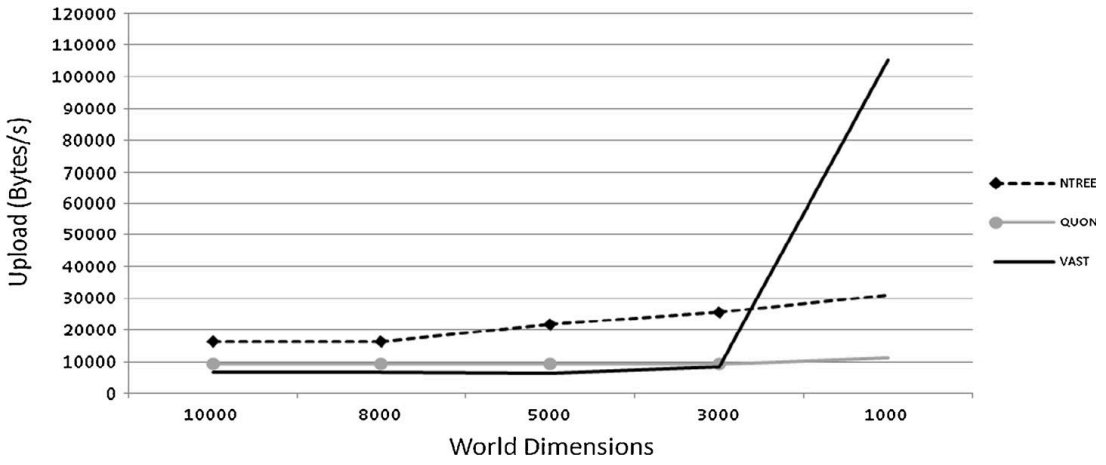


FIG. 2

Figure 3 [11] below depicts the average movement delay faced by clients in the network. This is the time it takes for a player to see someone moving near them in the virtual world and for their clients to respond to the movement. There is no discernible difference in delay between 10,000 by 10,000 and 1,000 by 1,000 area for VAST and QuON. The latency between the nodes is nearly the same because the two protocols retain direct communication with nodes in the area of interest. The two also seem to have consistently short delay periods. However, with N-Tree there is a clear gap because it tries to keep latency small by placing nodes in the network close to one another, the same as if they were in the virtual world. Its ability to scale heavily populated areas shows a flaw and the use of bandwidth will be too much, causing lag.

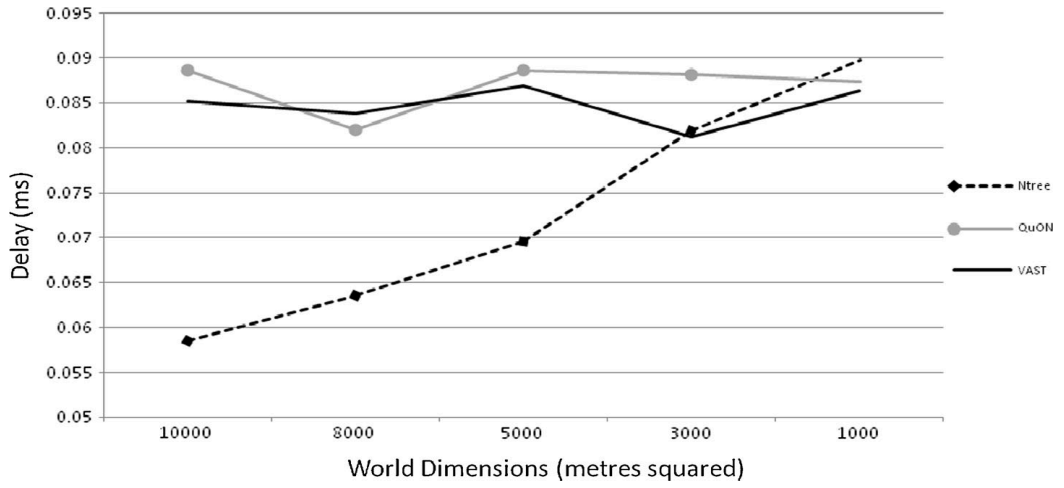


FIG. 3

Figure 4[11], depicts the typical travel delay faced by clients in the network in a crowded area test. It is important that protocols will handle the game's busiest area without players being constrained by latency. The N-Tree's delay increases to around 32 and 128 players but after that the results become unreliable due to network failure. Direct relation keeps delay in QuON and VAST to a minimum but to keep delay short both networks make bandwidth sacrifices.

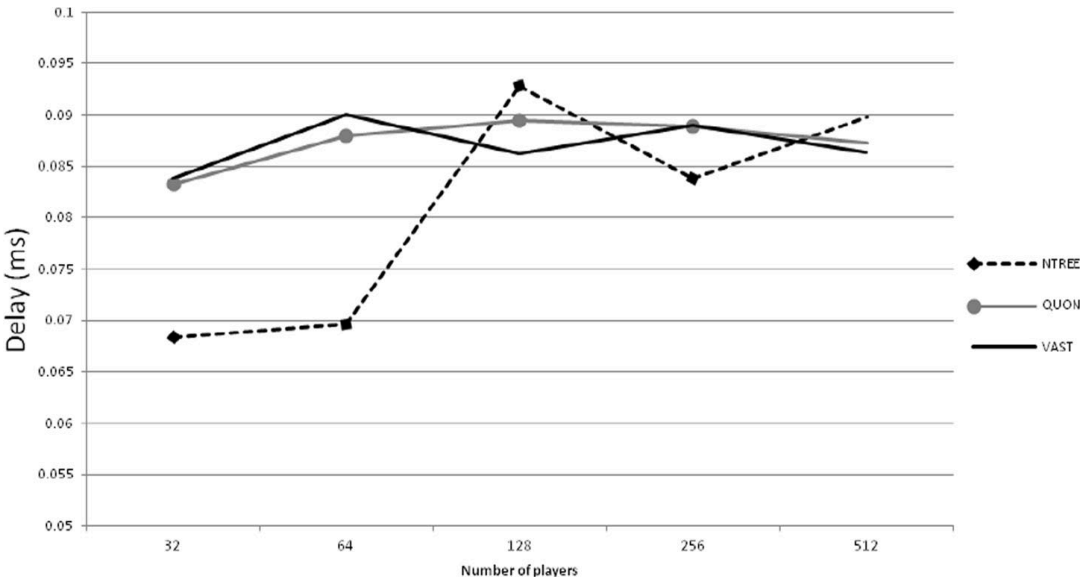


FIG. 4

From this paper, N-Tree and VAST scalability is not comparable to QuON. QuON's advantage comes from the limited number of direct contacts it employs. As long as there is adequate bandwidth for the protocols to use, VAST and QuON will still have low latency. Although VAST would fit well in a game with players spread across the virtual world, it is not realistic in MMOGs because player engagement is what distinguishes these games from other genres.

Conclusions

This paper offers an overview of the history of P2P, commonly used network architecture for MMOGs, some proposed approaches and protocols for P2P network in MMOgs and performance analysis. Recently, some P2P protocol is used in MMOGs locationally but still client server architecture dominates the field. However, we showed that P2P networks can keep low latency which is essential for MMOGs in a variety of situations. In addition, P2P does not require a huge central server, so the game session is flexible and scalable. Therefore, individual or indie game developers can make huge MMOGs without costing a lot of money for a server, so the development of MMOGs will be open to anybody by using the P2P network.

References

[1] Joshi, R., Patel, D., & Naik, S. (2016). Implementation of Peer-To-Peer Architecture in MMORPGs.

[2] Jiang, X., Safaei, F., & Boustead, P. (2007). An approach to achieve scalability through a structured peer-to-peer network for massively multiplayer online role playing games. Computer Communications, 30(16), 3075-3084.

[3] Gibson, M. S., & Vasconcelos, W. W. (2019). A knowledge-based approach to multiplayer games in peer-to-peer networks. Knowledge and Information Systems, 61(2), 1091-1121.

[4] Neumann C, Prigent N, Varvello M, Suh K (2007) Challenges in peer-to-peer gaming. ACM SIGCOMM Computer Communication Review 37:79–82,

[5] M. Pullen, M. Myjak, and C. Bouwens. Limitations of Internet protocol suite for distributed simulation in the large multicast environment, Feb. 1999. Request For Comments 2502.

[6] N. E. Baughman, M. Liberatore, and B. N. Levine. Cheat-proof playout for centralized and peer-to-peer gaming. IEEE/ACM Transactions on Networking. To appear.

[7] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), November 2001.

[8] GauthierDickey C, Lo V, Zappala D (2005) Using N-trees for scalable event ordering in peer-to-peer games. Proceedings of the international workshop on Network and operating systems support for digital audio and video. pp 87–92. doi: 10.1145/1065983.1066005

[9] Krause S, Backhaus H (2010) QuON – A quad-tree based overlay protocol for distributed virtual worlds. International Journal of Advanced Media and Communication 4:126–139. doi:10.1504/IJAMC.2010.032139

[10] Guibas L, Stolfi J (1985) Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi Diagrams. ACM Transactions on Graphics (TOG) 4:74–123. doi:10.1145/282918.282923

[11] Lu, L. et al (2013) 'Performance evaluation and simulation of peer-to-peer protocols for Massively Multiplayer Online Games', 74 (8):2763 Multimedia Tools and Applications

[12] Bjorn, K., Honghui L., Wei X., & Bryan H. (2004). Peer-to-peer support for massively multiplayer games

[13] Hanna, K., Emanuele, C., Laura, R., Alberto, M., & Massimo, C. (2015) Integrating peer-to-peer and cloud computing for massively multi-user online games. Peer-to-peer Networking and Applications 8, 301-319(2015)

[14] Yiting, X., Shah, N., & Dr. R.H. Mak. (2014) A Comparison of architectures in massive multiplayer online games. 2II45 Architecture of Distributed Systems Essay Type 3.

APPENDIX

Anthonese Mitchell, first semester as a MTIS student. She worked in the information technology field for a few years and is now looking to expand her knowledge in the security sector.

Haoyi Pan, fourth year computer science student. He has an interest in game development and has experience of working in a game company for a few months.

Kaimin Wang, fourth year computer science student, having some experience with researching for group projects of different aspects of technology. He also used to work on designing final project prototypes but not networking related.