Implications of Fog Computing on Multiplayer Games

Aaron Smith

Background: Interest in VR

- I’m interested in exploring multiplayer VR environments (HTC Vive and Rift)
- Oculus Rift: Substantially decreased the cost of consumer VR devices (2012)
- HTC Vive: Introduced “room scale” VR to consumers (2016)
    - Uses infrared “Lighthouse” cameras to track the headset and two controllers in 3D space

Background: Unique Properties of VR

- Higher frame rate requirements than 2D games (90fps vs 30-60fps)
- Position & orientation of the player is always changing (every single frame!)
    - More position updates sent over the wire than even the most fast paced first person shooter (2 hands and a head, each with position and rotation)

Background: Why VR and not AR?

- VR is a more mature platform
- Many UI/UX properties of VR applications are easily transferrable to AR
  - Menus as “physical objects” instead of floating 2D menus
  - Similar solutions for interacting with objects in both mediums
- Cost
  - HoloLens = $3000
  - HTC Vive (“premium” VR experience) = $800

Constraints

- Application must support multiple players in a world with shared state
- Application must be able to maintain 90 fps over the life of the game
- Application must be able to guarantee 90 fps for a to be determined number of players
- Data: Using test application
  - When one player interacts in the VR application for 5 minutes, there is an average of ___ kb of data generated per frame that must be sent over the wire
    - TODO: (position (Vector3) + rotation (Vector3) + other data?) * 3
      - 3 = headset + left controller + right controller
      - More data than that will be sent when other objects are interacted with
  - This large amount of data going over the network makes it challenging to keep real time requirements

Problem

What real time and frames per second guarantees can be made for a multiplayer VR application using a novel peer to peer model based on state sharding?

Test application

- Unity created environment with a cube and sphere that can be picked up and thrown around by VR players
- Can be viewed by using a VR headset or a computer
  - Only VR headset can interact with the environment
- VR player can throw items, other VR players can catch them
- Computer player can “catch” items by pressing a key

Existing Solutions

- Client-server
  - Clients send their “actions” to the server, which reconciles those actions and distributes the updated world state back to the clients
- Peer-to-peer
  - Clients send their “actions” to each other
  - Voting mechanisms can be used to resolve conflicts and catch cheating
  - Global state must periodically be reconciled amongst all peers (compare hashes)
- Client-server with distributed servers
  - Clients send their “actions” to the closest geographical server
  - State is partitioned amongst all servers and uniquely identifiable by GUIDs
  - “Primary copies” of objects are replicated into “secondary copies” to improve performance
  - Actions are forwarded to server which owns primary copy for execution

Proposal: Peer-to-peer solution with state sharding

- Use the concepts described in A Distributed Architecture for Interactive Multiplayer Games in a peer to peer context
- Identify each object by a GUID
- Keep a primary copy of each object on one node in the system
  - This is where all updates are performed
- Keep secondary copies on other nodes in the system as necessary to reduce network traffic
- Use execution partitioning to execute think functions for only primary copies of objects owned by a node
- Forward updates from secondary copies to the primary copy
- Forward delta encoded updates from the primary copy to all replicas
- Delete replicas if no updates arrive

Proposal: Performance Placement

- As a separate part of this project, Shashank will research secondary copy prediction and location optimization

Cheating: Overview

- 5 important protocol-level cheats in P2P applications
  - Fixed-Delay Cheat, Timestamp Cheat, Suppressed Update Cheat, Inconsistency Cheat, Collusion Cheat
- In “Cheat-Proof Playout for Centralized and Distributed Online Games (2001)”, the “Lockstep protocol” is introduced
  - There is a 3x negative performance impact
  - Only solves Fixed-Delay and Timestamp cheats
- In “Low Latency and Cheat-proof Event Ordering for Peer-to-Peer Games”, the authors introduce the “New-Event Ordering (NEO)” protocol, which improves on the Lockstep protocol
  - All 5 protocol level cheats are addressed
  - 2 * round length performance impact when using pipelined approach

Number of Players

- This study looks at incremental scalability; it is not an exercise in handling the largest number of players possible, like in a MMORPG
  - We hope to answer questions like: How is my performance impacted as more players join?
- Given a lack of available hardware and time, this study will focus on the incremental scalability from 1 to 8 players

Area of Interest (AOI)

- Optimizations to AOI are most important in MMORPGs
- For this project, assume a single AOI
- A large part of the existing literature focuses on the problem of MMORPGs and produces solutions based on optimizing the AOI
  - VoroGame : A Hybrid P2P Architecture for Massively Multiplayer Games
  - A Distributed Architecture for Interactive Multiplayer Games

Strategy

- Create a library for multiplayer games that can be used with any Unity game without modification
  - Use the architecture proposed in Matrix: Adaptive Middleware for Distributed M Games to keep a clean separation
- Study the primary/secondary copy sharding approach described in previous slides
- Measure network traffic and real time performance with other approaches (single server, standard p2p)
- Determine if all five protocol level cheats can be protected against using the NEO protocol (do not implement)

Results

In order to measure the performance of my solution, I will use the following tools:

- Built-in Unity profiler for CPU/GPU usage, memory, and frame rate profiling
- Wireshark for packet sniffing to measure average network traffic and incremental traffic when new users are added