Project Matryoshka: NDN Multiplayer Online Game

Zhehao Wang[1], Zening Qu[2], Jeff Burke[3]
[123]UCLA REMAP

Introduction

Matryoshka is a peer-to-peer multiplayer online game (MOG) running on Named Data Networking (NDN).
We identify the MOG synchronization problem, then propose an octree partition of the game world, and a two-step synchronization design.

Background

Peer-to-peer structures were explored for online games.
Maintaining security and availability while scaling users has driven most MOGs towards a client-server or client-superpeer architecture.
Client-server games face certain problems:
1. a small number of points of failure;
2. traffic centralizing at servers.
To tackle these problems, we present Matryoshka, whose design is based on Sync: by exchanging data digest each party learns about the missing data, and then can retrieve data via built-in multicast delivery.
The demo demonstrate Matryoshka's gameplay.

Gameplay

- Play as a matryoshka
- Explore unknown universe
- Discover other players
- Get player position updates
- Interact with environment

Figure 1: Game play

Problem Analysis

The challenge of the MOG: peer-to-peer synchronization in a distributed virtual environment.

- Synchronization
  Players whose areas of interest intersect with each other should reach consistent conclusions about things in the intersected area.
- Locality
  A player needs to know the updates of objects within its Area of Interest (AoI) in the virtual game world.

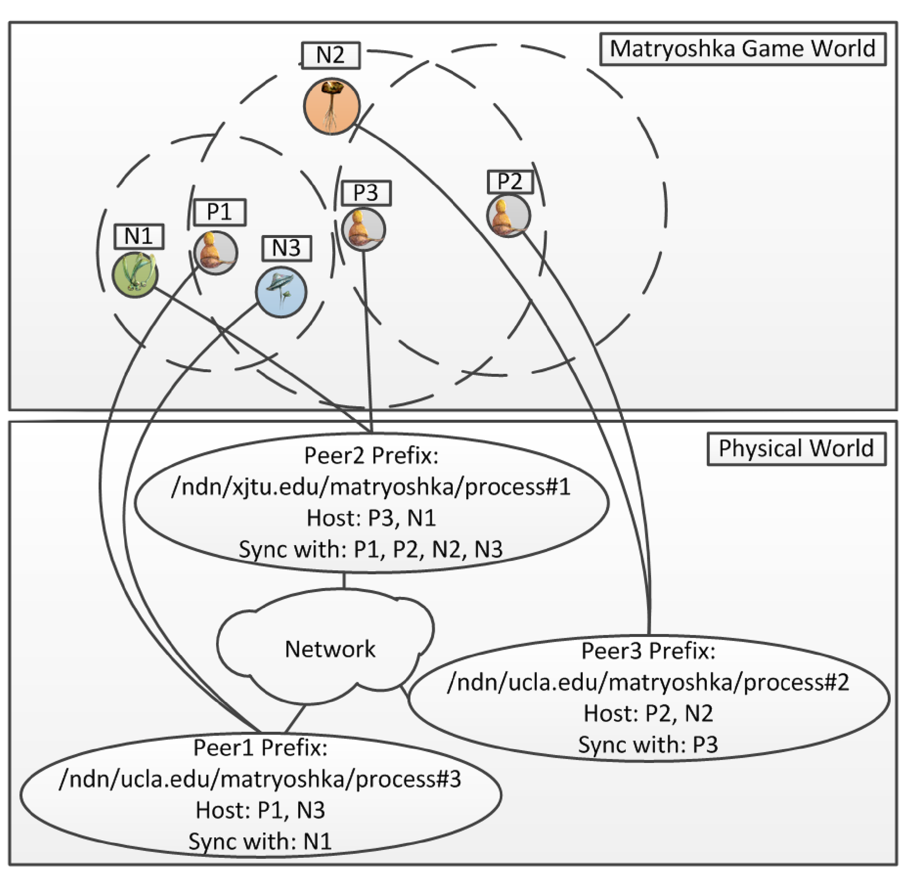

Figure 2: Two worlds in the MOG

Figure 2 illustrates the problem.
Each physical peer hosts a player. Take peer 2 as example.
- It hosts player3.
- It should discover player1, player2, NPC2 and NPC3.
- Its knowledge of player1 and 2's locations should be updated.
- The AoI should move as player3 moves.

Virtual World Partitioning

Figure 3 illustrates the octree partition of the virtual environment.
The whole world is represented by a top-level cube.

- Octree partitions the virtual environment into octants statically and recursively.
- With octree, we provide a shared namespace: all the peers that care about the same region can share the data brought by synchronization interests.

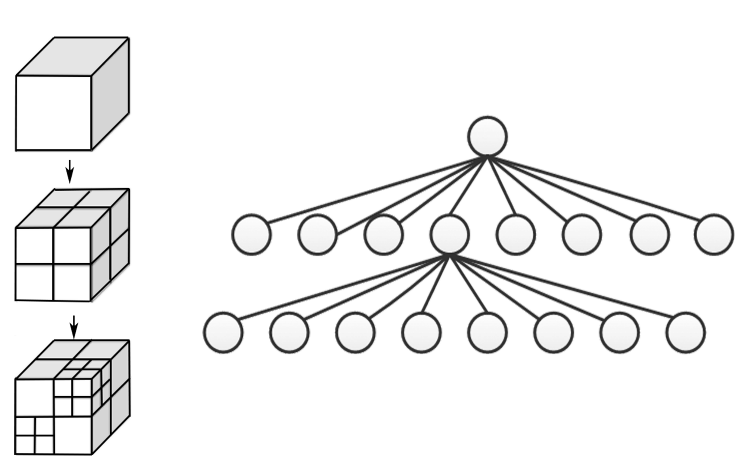

Figure 3: Octree partitioning

Contact Information

- Email: wangzhehao410305@gmail.com

Design Overview

Two-step design:
- **Discovery**: which players are in a peer's vicinity.
  Peers with overlapping AoIs synchronize "discovery namespaces" for octants of mutual interest to find other objects in the game world:
  - They periodically express Interests in the discovery namespace containing the octant indices they are interested in to all peers, along with a digest of the object names they know in each octant.
  - Each peer responds with their knowledge of objects in the octants.

  The namespace for discovery is given in Figure 4.
  - Game name component separates the game into sub-worlds.
  - Octant indices indicate the octants absolute location in the game world.
  - Digest component contains the hash of the set of object names.

  All peers have the same hash for octants belonging to their intersection when steady state is reached.

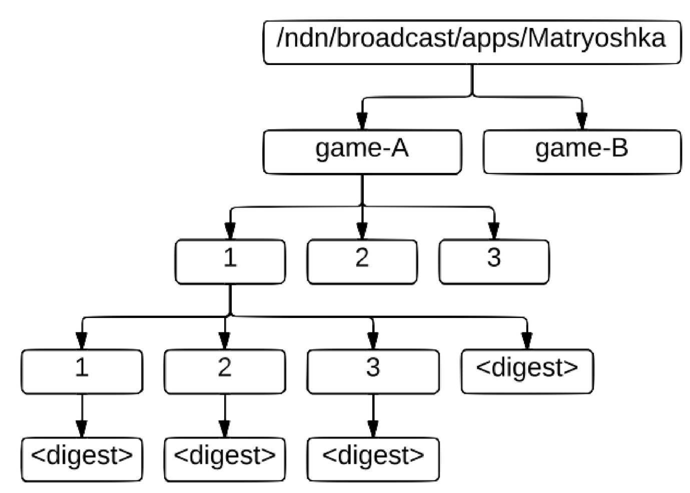

Figure 4: Discovery namespace

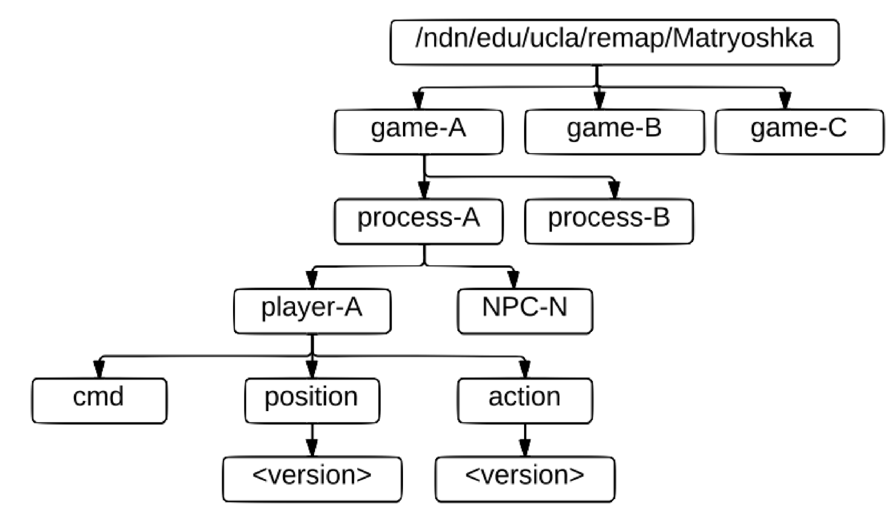

Figure 5: Update namespace

- **Update**: what are players doing.
  - Peers express position update interests using the object names returned in step one.
  - Peers publish their location with sequence numbers.

  Figure 5 illustrates the update namespace.
  - Process name represents a game instance, which hosts the player avatar.
  - Position and action components fetch the latest data.

Pattern of communication for a game instance (a) to discovery another instance (d) is given in Figure 6.

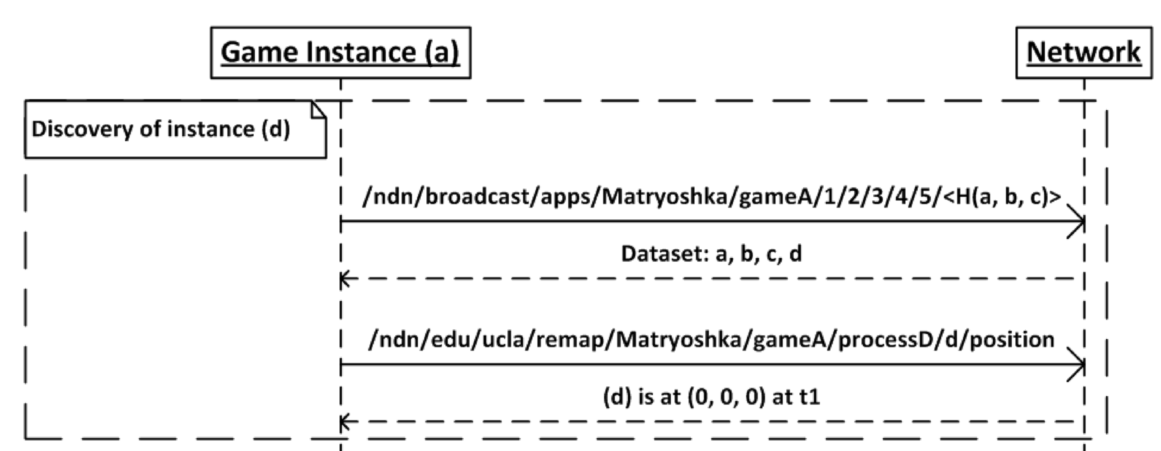

Figure 6: Example names

Demo Implementation

The demo application was built using Unity3D game engine, and ndn-dot-net, a C# adaptation of NDN Common Client Library. The demo shows the game code running on a small number of peers, and a visualization of the AoI.
Player characters and NPCs are instantiated by each peer, and each peer can navigate around the common world using their player character. Player and NPC discovery, and player position update (at a rate of 25 Hz) is demonstrated.

Future Work

Future work includes further evaluation of the design, and addressing problems caused by hierarchical octree partitioning.
- Difficult to represent AoIs that are close to the border between two sub-regions of the highest subdivision hierarchy.
- Difficult to represent spherical AoI.
- Need routing to keep up with the changes in AoI represented by octree.

References

[1] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael Plass, Nick Briggs, and Rebecca Braynard. Networking named content. *Commun. ACM*, 55(1):117–124, January 2012.
[2] Zhenkai Zhu, Chaoyi Bian, Alexander Afanasyev, Van Jacobson, and Lixia Zhang. Chronos: Serverless multi-user chat over ndn. Technical Report NDN-0008, NDN, October 2012.

 zhehao.mail@gmail.com