Recomendador de equipos para juegos MOBA usando filtro colaborativo

Karen Gordillo Viña, Alexis Mendoza Villarroel

May 2019

1 Introducción

Los videojuegos online de batalla en arena(MOBA) se han convertido en uno de los generos mas jugados en recientes anios. Este es el caso de *Defense of the Ancients* (DOTA), *League of Legends* (LOL), *Heroes of the Storm* (HOTS), entre otros. En estos juegos usualmente se encuentran dos equipos de 5 jugadores cada uno, cada jugador controla una unidad (heroes o campeon), que se diferencia de las demas por sus caracteristicas y habilidades. Cada equipo tiene la mision de derrotar las unidades y estructuras enemigas.

Al inicio de cada partida, cada jugador puede escoger un heroe de entre muchos. Como cada una de estas unidades posee diferentes habilidades, fortalezas y roles, es fundamental elegir una buena combinacion de equipo para ganar. Por tanto en este trabajo propones un sistema de recomendacion de equipos. Debido a la disponibilidad de data elegimos usar un sistema de filtro colaborativo, estos sistemas se basan en la similitud entre usuarios o *items*, y la colaboracion de ellos para realizar recomendaciones personalizadas. En [1] se realizo un trabajo similar recomendando lineas de heroes.

2 Marco Teórico

2.1 Sistemas de recomendaciones

Los sistemas de recomendación son herramientas de *software* y técnicas que proveen sugerencias de *items* a ser usados por un usuario [**?**], donde un *item* es el termino usado para denotar lo que el sistema recomienda a los usuarios. Los sistemas de recomendación cumplen con dos tareas principales, predicción y recomendación [**?**]. La predicción es usada para saber la valoración que un usuario le daría a un nuevo *item*. Mientras que la recomendación se encarga de recomendar una lista de *items* que a un usuario le podría gustar. Los sistemas de recomendación usan varias fuentes de información ya sean implícitas o explicitas para generar sus recomendaciones. Los algoritmos de recomendación se categorizan en tres grupos: los basados en contenidos, lo filtros colaborativos y

los algoritmos de recomendación híbridos.

2.1.1 Filtro colaborativo basado en memoria

Los algoritmos de filtro colaborativo basados en memoria usan la total o parcial base de datos de valoraciones para generar nuevas predicciones [**?**]. Cada usuario es parte de un grupo con intereses similares, formando una vecindad. Identificando estos vecinos es posible predecir las preferencias de un usuario.
Este algoritmo basado en vecindad consiste en los siguientes pasos: calcular la similitud o peso w_i, j, que refleja la distancia entre dos usuarios o *items*; producir la predicción para el usuario tomando el peso ponderado de todos las valoraciones del usuario o *item* en un cierto usuario o *item* como se describe en [**?**]. Este algoritmo de divide en dos categorías la basada en usuarios y la basada en *items*.

Filtro colaborativo basado en usuarios Esta técnica se encuentra entre mas exitosas y ampliamente usadas. Se usa para recomendar un conjunto de *items* a un usuario basándose en las valoraciones de otros usuarios similares. El primer paso es encontrar la vecindad del usuario. Luego se estima la nueva valoración considerando los pesos de cada vecino, donde los pesos reflejan la similitud del usuario y sus vecinos. Por tanto entre mayor sea la similitud del usuario a un vecino, mayor sera el impacto de la valoración del vecino en la estimación de la nueva valoración para el usuario. La nueva valoración para el nuevo usuario u con el item i se calcula usando:

$$r_u i = \frac{\sum\limits_{u' \in NS_u} (r_{u'i}) x (similitud(u, u'))}{\sum\limits_{u' \in NS_u} |(similitud(u, u')|} \tag{1}$$

3 Propuesta

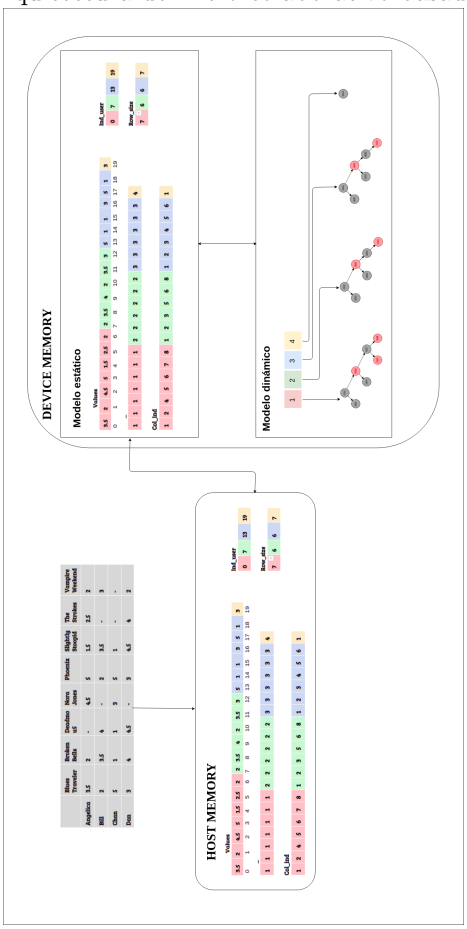
Nuestra propuesta consiste en usar filtro colaborativo basado en usuarios, para recomendar un equipo completo, dado un equipo incompleto. En la siguiente seccion se discutira la arquitectura a usar para el filtro colaborativo basado en usuarios.

3.1 Arquitectura

Debido a la naturaleza de los datos de estos juegos, es necesario tener una arquitectura capaz de manejar grandes volumenes de datos, ser actualizada dinamicamente y ser rapida para realizar recomendaciones antes de que se inicie un juego. Para ello nuestra propuesta incluye una representacion esparsa de la matriz de *items*, el uso de arboles *red-black* para permitir actualizaciones instantaneas de la data, ademas del uso del poder computacional de la GPU para realizar recomendaciones en tiempo real.

Nosotros consideramos el uso de un modelo estatico y uno dinamico para el manejo de los datos. El modelo estatico consiste de una representacion esparsa de la matriz de valoraciones de *items*, en ella se separa la matriz en tres arreglos de un mismo tamanio, por cada posicion i en los arreglos se almacena de la fila y el el valor, el idid de la columna respectivamente. El modelo estatico es usado para realizar la creacion del modelo dinamico en GPU. Una vez creada el modelo dinamico este es usado en vez del estatico para actualizar y realizar recomendaciones. Nuestro movelo dinamico consiste en un vector de cuda thrust que almacena punteros a arboles *red-black*. Cada posicion en este vector representa los ids de los usuarios, y el arbol asociado a cada posicion en el vector representa un arbol balanceado de los items y respectivas valoraciones del usuario. La arquitectura puede verse en detalle en la figura 1.

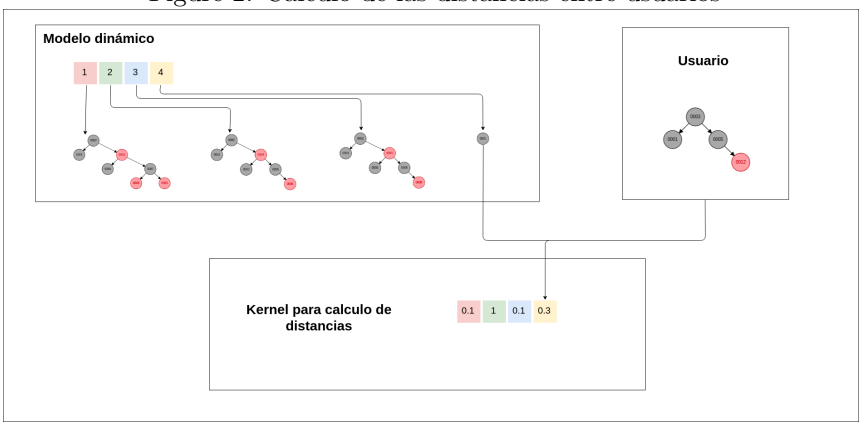
Figure 1: Arquitectura del filtro colaborativo basado en usuario



Para realizar las recomendaciones, se llaman kernels que realizan los calculos

en el modelo dinamico, dado un usuario. Estos kernels calculan distancias entre los usuarios y el usuario objetivo, almacenando los resultados en un arreglo en memoria del device. EL proceso puede visualizarse en la figura 2.

Figure 2: Calculo de las distancias entre usuarios



3.2 Recomendacion de equipos

Para realizar las recomendaciones consideramos a los equipos como usuarios y a sus heroes como *items*. Usando este enfoque es irrelevante las valoraciones por lo que se usa un numero constante para todos. Para realizar las pruebas consideramos el dataset de LOL, el cual mantiene informacion de partidas clasificatorias por temporadas. De la data provista obtuvimos por cada partida los ids de los heroes por cada equipo. Luego usamos estos datos para inicializar nuestro modelo estatico y dinamico. Una vez inicializados usamos un kernel en cuda para realizar la distancia de pearson entre un equipo incompleto dado y todos los equipos registrados. De esta forma es posible encontrar equipos con heroes similares a los que se esta eligiendo. permitiendo asi realizar recomendaciones para los heroes faltantes del equipo.

4 Resultados

El sistema de filtro colaborativo basado en usuario logro hacer recomendaciones en tiempo real en 0.1 ms;

5 Conclusiones

En este trabajo se presenta un sistema de recomendacion para recomendar heroes durante el proceso de seleccion de heroes usando la data provista por

Oracle's Elixir. El sistema alcanza su objetivo recomendando buenas composiciones de equipo que permitan ganar partidas. El uso de CUDA en los calculos de las distancias permite alcanzar el objetivo de realizar recomendaciones en tiempo real, ademas el uso de un arbol *red-black* facilita la modificacion de valores dentro de los items de cada usuario. Como trabajo futuro queremos realizar las recomendaciones de equipos basandose tanto en el equipo aliado como en el equipo enemigo. Es tambien posible realizar la prediccion del equipo jugador usando un filtro colaborativo con pesos. Otra forma de realizar predicciones es el uso de redes neuronales, que podrian usar una mayor cantidad de data a la usada aqui sin necesidad de preprocesamiento.

6 Referencias

References

[1] L. Hanke and L. Chaimowicz, "A recommender system for hero line-ups in moba games," in *Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2017.