Final Project Proposal

Team Member

Foo Jia Yin (106062361)

Target Topic

Parallelism and Acceleration in Recommendation Systems

Motivation

In the era of big data, recommendation system is widely utilized in from commercial e-shops to social networks, product review sites other SaaS applications. Many websites collect user profiles to provide some valuable information through personalized recommendation.

Due to the increasing scale of data in these applications is constantly increasing, in some practices, the computational time of recommendation system algorithm may be unsatisfying. Different approaches have been proposed to deal with the problem of scalability of recommender systems'algorithms. This project will focus on implementation of parallelism approaches with C++ and compare the computational time with.

Recommendation System: Overview

There are three major types of recommendation system algorithm: Content-based algorithm, Collaborative Filtering (CF) algorithms and model-based algorithms. Below listed the pros and cons for different types of algorithm to show their importance.

1. **Content-based Algorithms**

   Algorithms:
   - TF-IDF

   Pros:
   - No need for data on other users
   - Able to recommend to users with unique tastes
   - Able to recommend new & unpopular items

   Cons:
   - Need different features for text, image, music…
   - Recommendations for new users
   - Overspecialization

2. **Collaborative Filtering (CF) Algorithms**
   - User-user CF
   - Item-item CF

   Pros:
   - Works for any kind of item

   - Works better for multiple taste user

   Cons:

   - Cold start: Need enough users in the system to find a match
   - Tends to recommend popular items

3. **Model-based**
   - Latent Factor Model
   - Auto-encoder Model

This project involved the following task:

1. Acceleration on **Collaborative Filtering**
2. Acceleration on **Content-based** Algorithm (advanced, if have time)

Performance Evaluation

Dataset: https://grouplens.org/datasets/movielens/

Use different scale of data to compare the computational time of recommendation algorithm

1. With parallelism
2. Without parallelism

Data scale:

1. Tiny: **22** ratings applied to **5** movies by **5** users
2. Small: **100,000** ratings applied to **9,000** movies by **600** users
3. Large: **27,000,000** ratings applied to **58,000** movies by **280,000** users

Advanced: Compare performance under below situation:

1. Sparse ratings vs dense ratings
2. More user vs more movies

Schedule

Week	Date	Task
1	5/19~5/22	Paper study on different parallel architecture on collaborative filtering
2	5/23~5/29	Implement Collaborative Filtering in C++ without parallelism
3	5/30~6/5	Implementation of parallelism – Algorithm 1
4	6/6~6/12	Implementation of parallelism – Algorithm 2 (advanced)
5	6/13~6/19	Performance evaluation and Optimization
5	6/13~6/19	Performance evaluation and Optimization

Implementation Plans

Algorithm to accelerate: Item-item collaborative filtering

Stage 1: Data standardization

- For each user-item, substract the rating by its mean
- Treat missing data as 0

Stage 2: Calculate magnitude = ||r|| for each movie

$$||r_x|| = \sqrt{\sum_{i\in N} \boldsymbol{x}_i{}^{\boldsymbol{2}}}$$

Stage 3: Calculate Item-item similarity
for each movie pair (m1, m2), calculate their cosine similarity

$$\text{sim}(\boldsymbol{x}, \boldsymbol{y}) = \cos(\boldsymbol{r}_x, \boldsymbol{r}_y) = \frac{r_x \cdot r_y}{||r_x|| \cdot ||r_y||}$$

Stage 3: Calculate predicted rating for user-item

$$r_{xi} = \frac{\sum_{j\in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j\in N(i;x)} s_{ij}}$$

Parallel Algorithm 1: My algorithm

- Use map-reduce in different stage

Parallel Algorithm 2: algorithm in latest paper which I can understand (advanced)

- To be studied

Reference

1. Zhou Y., Wilkinson D., Schreiber R., Pan R. (2008) Large-Scale Parallel Collaborative Filtering for the Netflix Prize.
2. Karydi, E., & Margaritis, K. (2016). Parallel and distributed collaborative filtering: A survey.
3. Ali, M., Johnson, C. C., & Tang, A. K. (2011). Parallel collaborative filtering for streaming data.
4. Jiang, J., Lu, J., Zhang, G., & Long, G. (2011). Scaling-up item-based collaborative filtering recommendation algorithm based on hadoop
5. Muqeet Ali, Christopher C. Johnson, and Alex K. Tang. 2011. Parallel Collaborative Filtering for Streaming Data.

6. Panigrahi, S., Lenka, R. K., & Stitipragyan, A. (2016). A hybrid distributed collaborative filtering recommender engine using apache spark. Procedia Computer Science, 83, 1000-1006.
7. Sun, Jiankun & Wang, Ziyang & Luo, Xiong & Shi, Peng & Wang, Weiping & Wang, Long & Wang, Jenq-Haur & Zhao, Wenbing. (2020). A Parallel Recommender System Using a Collaborative Filtering Algorithm With Correntropy for Social Networks
8. Sardianos, Christos & Papadatos, Grigorios & Varlamis, Iraklis. (2019). Optimizing Parallel Collaborative Filtering Approaches for Improving Recommendation Systems Performance.