Session 26: Recommender Systems

担当 駒水、上江、山口 （筑波大）

2011/12/10 VLDB2011勉強会

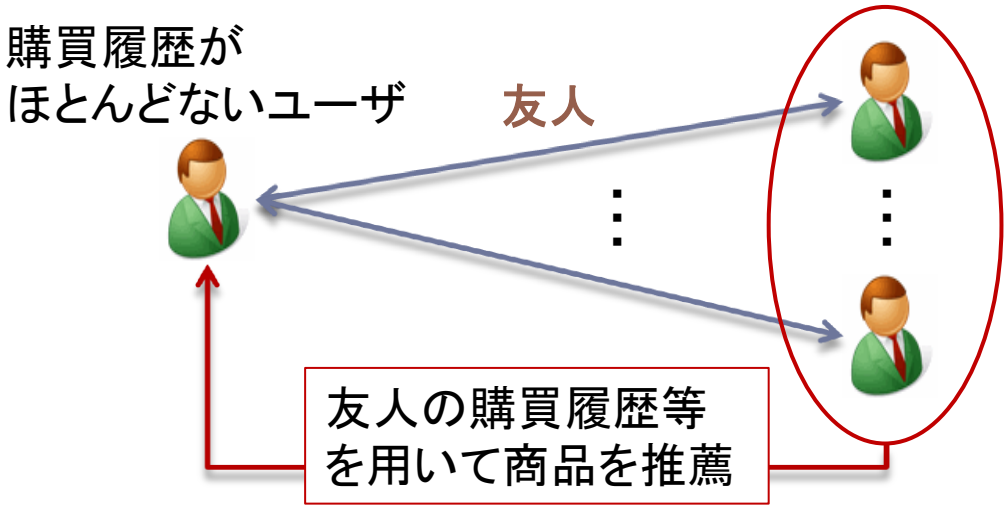
Session 26: Recommender Systems

- **Personalized Social Recommendations – Accurate or Private?**
  - 担当：山口
- **RecBench: Benchmarks for Evaluating Performance of Recommender System Architecture**
  - 担当：駒水
- **MRI: Meaningful Interpretations of Collaborative Ratings**
  - 担当：上江

1. Presonalized Social Recommendations – Accurate or Private?

背景

Facebookなどのソーシャルネットワークの普及
➔ソーシャルネットワークを利用した推薦アルゴリズムが実現可



友人のプライバシー侵害になり得る！

1. Presonalized Social Recommendations – Accurate or Private?

推薦精度を高めるために多くの情報を用いるとプライバシー侵害の可能性大

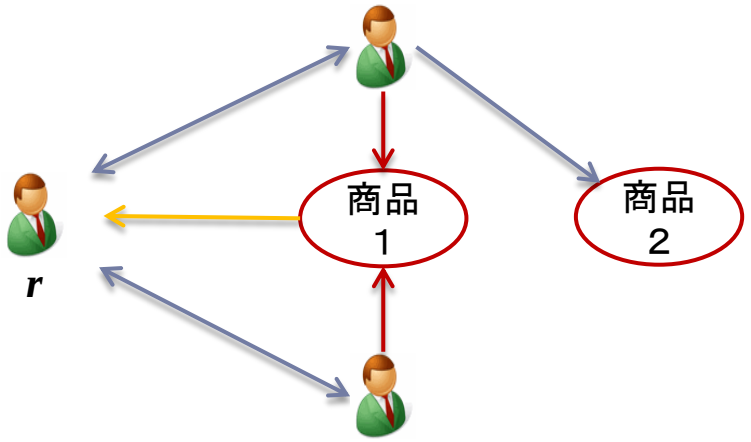
プライバシーを保護するために情報を隠すと推薦精度が低くなる

研究目的

プライバシーと推薦精度のトレードオフを定量的に分析する

1. Presonalized Social Recommendations – Accurate or Private?

問題定義（１/３）



Social Recommendation Algorithm

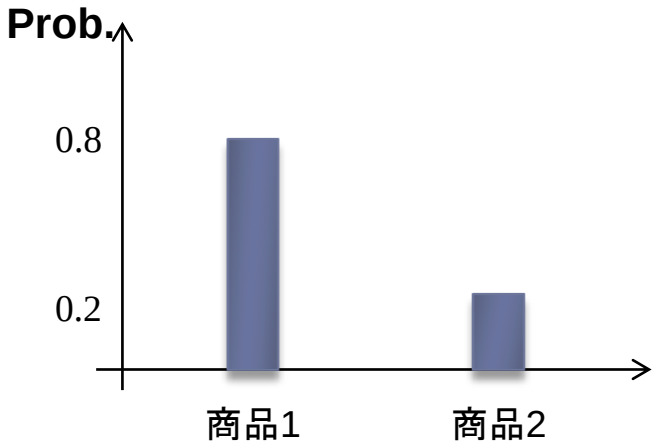
入力：ソーシャルグラフ　G = (V, E)
V はユーザ及び商品などのノード
E は友人関係や、商品購入など

出力：エッジ (i, r)
ターゲットユーザ r に対してノード i を推薦することを意味する

アルゴリズム：r に対して i を推薦する確率 pi の分布で表す

通常は r に対して **utility** uiが最も大きい i を確率 1 で推薦するが、プライバシー保護のため**確率分布として ”ぼやかす”**

utilityの例）number of common neighbors
sum of weighted path

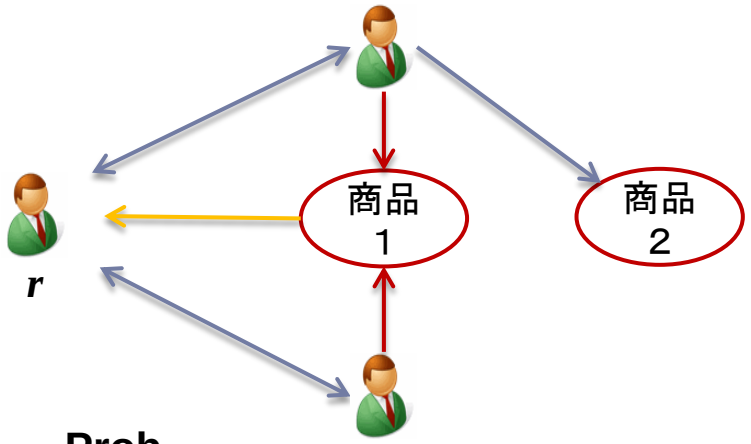


1. Presonalized Social Recommendations - Accurate or Private?

問題定義（２/３）

Accuracy

$$\min_{\mathbf{u}} \frac{\sum u_i p_i}{u_{\max}}, \quad u_{\max} = \max_i u_i$$

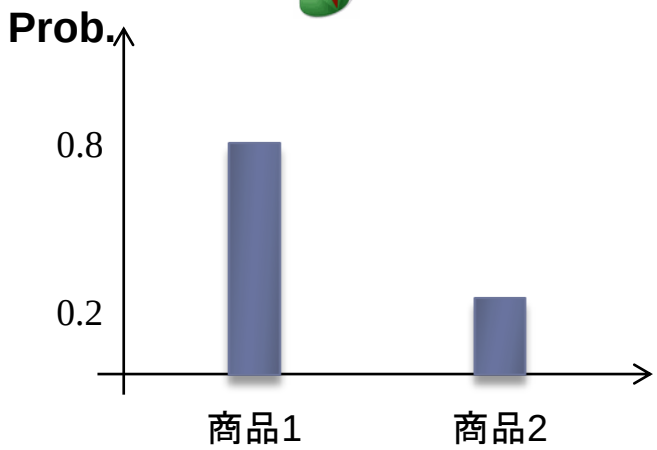


例）utility: # of common neighbors

$$u_1 = 2, \quad u_2 = 1$$

$$p_1 = 0.8, \quad p_2 = 0.2$$

$$accuracy = \frac{2 \cdot 0.8 + 1 \cdot 0.2}{2} = 0.9$$



ぼやかすほど精度は悪くなる！

1. Presonalized Social Recommendations - Accurate or Private?

問題定義（３/３）

ε-differential privacy（ε-差分プライバシー）　　e.g. G' = G + {e}

グラフ G と、エッジを一つ編集して得られるすべてのG' に対して推薦アルゴリズム R を適用し

$$\frac{\Pr[R(G) \in S]}{\Pr[R(G') \in S]} \leq e^{\varepsilon}$$

R(G) ： R による推薦結果（一つのエッジ）
S　　： 取りうる推薦結果のエッジの集合
ε　　： 小さい正数

であるとき、ε-differential privacyが満たされているという

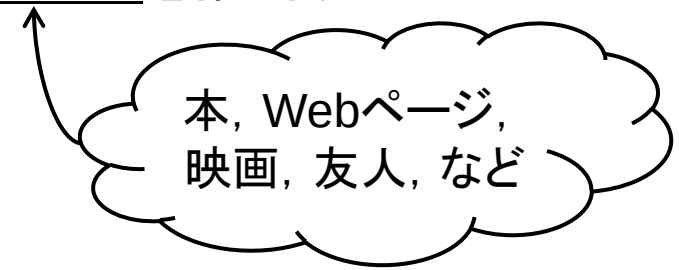
* 確率の比がほぼ1のとき

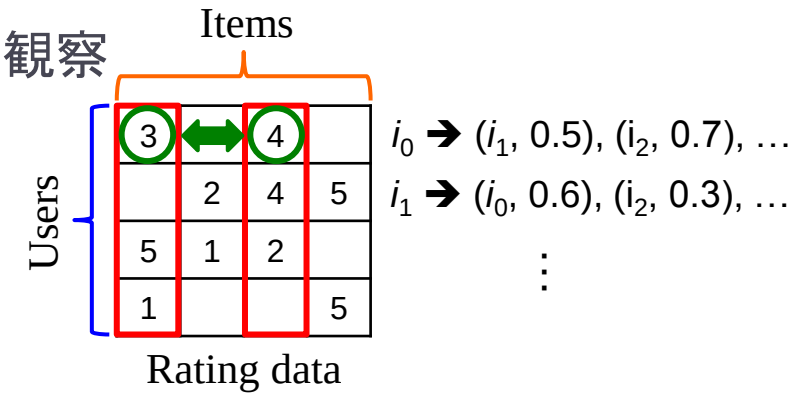
ε-differential privacyを満たしつつ、Accuracyを最大にするSocial Recommendation Algorithmを構築するという問題

続き（証明たくさん）は論文で…

2. Levandoski et al., RecBench: Benchmarks for Evaluating Performance of Recommender System Architectures

- 情報推薦：ユーザの興味に合いそうなアイテムを推薦するシステム
- 情報推薦システムの分類
  - “手製”のシステム（MovieLens など）
  - データベースを利用したシステム
  - データベースに組み込むシステム（RecStore など）
- 疑問：各システムが実際にどの程度で実行できるのか？
- この論文がしたこと
  - アーキテクチャによるパフォーマンスの観察
- RecBench の前提
  - アイテムベースの協調フィルタリング
  - MovieLens*1 と Netflix prize*2 を利用





*1: MovieLens Datasets: http://www.grouplens.org/node/73, *2: Netflix Prize Dataset: http://www.netflixprize.com

2. Levandoski et al., RecBench: Benchmarks for Evaluating Performance of Recommender System Architectures

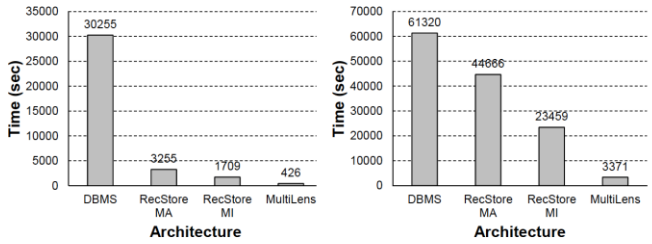
- ベンチマークタスク
  - Initialization
    - 情報推薦が可能になった状態
  - Pure recommend
    - ユーザに *n* 個のアイテム推薦
  - Filtered recommend
    - 条件付きのアイテム推薦
  - Blended recommend
    - 自由記述クエリと推薦の結合
  - Item prediction
    - ユーザの Rating を推定
  - Item update
    - 新しいアイテムを追加

データセット:
- MovieLens
  - 10M movie ratings
  - ~10K movies, ~70K users
- Netflix Challenge
  - 100M movie ratings
  - ~18K movies, ~480K users

評価:処理時間

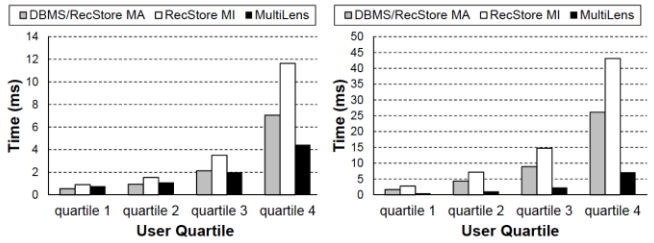
アーキテクチャ:
1. “手製”システム – MultiLens –
2. DBMS (PostgreSQL) を利用したシステム
3. 情報推薦をサポートするDBストレージエンジンを用いたシステム -RecStore-

2. Levandoski et al., RecBench: Benchmarks for Evaluating Performance of Recommender System Architectures



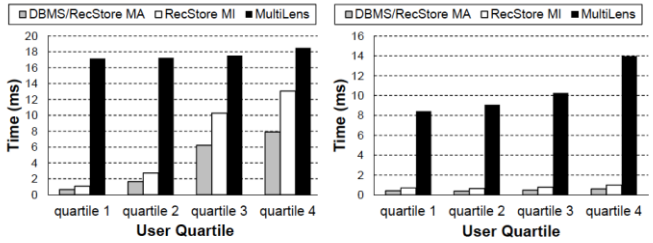
(a) MovieLens Data (b) Netflix Data

Figure 3: Initialization task.



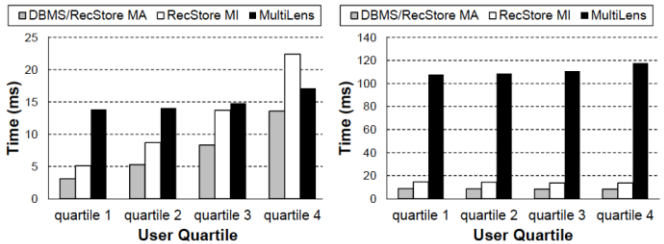
(a) MovieLens Data (b) Netflix Data

Figure 4: Pure recommend results.



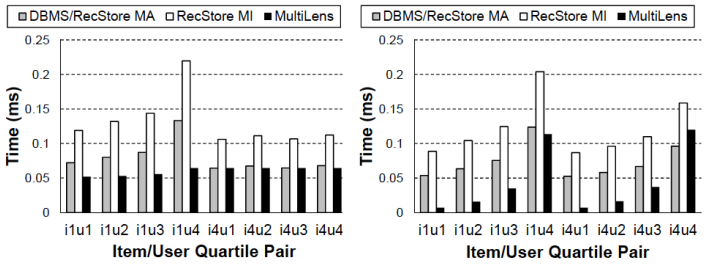
(a) MovieLens Data (b) Netflix Data

Figure 5: Filtered recommend results.



(a) MovieLens Data (b) Netflix Data

Figure 6: Blend recommend results.



(a) MovieLens Data (b) Netflix Data

Figure 7: Item prediction results.

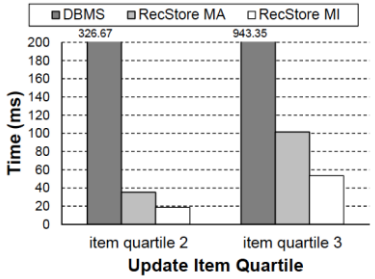


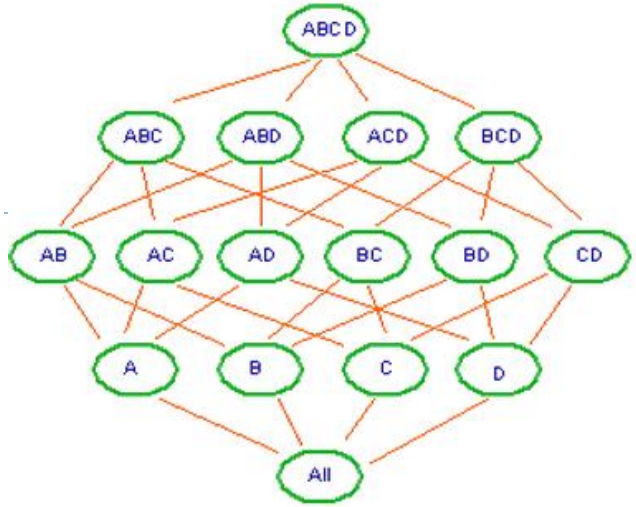
Figure 8: New item update results.

MRI:
Meaningful Interpretations of Collaborative Ratings

- 研究背景： レビューサイトの利用
  - 全てのレビューを読む ⇒ 時間がかかる
  - 評価値を見る ⇒ 詳細な情報を落としている

  アイテム集合に対する評価情報が得られない
    - アイテム集合： ある監督の映画に関する評価、等
- 目的： 評価情報を意味解釈して利用者に提示
  - 評価者とアイテムの属性を集合として利用
    - アイテム集合に対しても適用可能
  - 「10代の女性評価者はこのアイテムが好きだと言っています」
- アプローチ： DEMとDIM を検出して提示
  - DEM（Meaningful Description Mining）：
    アイテムに対して類似する評価値を持つ評価者集合を検出
  - DIM（Meaningful Difference Mining）：
    アイテムに対してグループ間で賛否両論となる評価者集合を検出

データモデルと問題定義

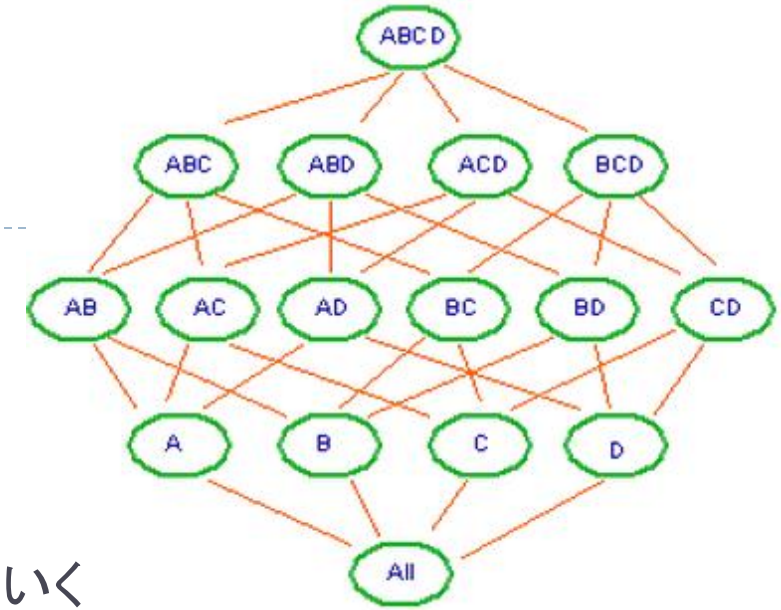


- データモデル
  - レビューサイト:
    ＜アイテム集合，評価者集合，評価集合＞
    - 評価:＜アイテム属性，評価者属性，評価値＞
  - グループ： アイテム/評価者の属性値が共通する評価集合
    - グループは格子構造になる
- 問題定義
  - DEM： 評価値の分散を最小化するグループ集合を探索
    - 条件： グループの数がk以下、coverageがα 以上
  - DIM： 評価値の差が激しいグループ集合を探索
    - 高評価と低評価、両方の評価を含むグループの割合
    - 条件： グループの数がk以下、両グループのcoverageがα 以上

coverage： アイテムに関する評価の内，グループがカバーしている評価の割合

DEM・DIM 共にNP-Complete ⇒ **効率的な手法を提案**

アルゴリズム



- DEM
  - 初期グループ集合 *G* :
    ランダムに選んだkグループ
  - coverageを満たすグループ集合を検出
    するまで*G* 中の1グループずつswapしていく
    - swap：格子上、線で直接つながっているグループに置換
  - 検出したグループ集合について、swapして生成可能なグループ内で
    coverageを満たしつつ評価の分散が最小になるグループ集合を返す
- DIM
  - グループ集合中の全てのグループの組合せについて、高/低評価を見つける必要があるので、計算コストが高くなる
    - Fundamental Regions という概念により効率化
      - 各評価毎にグループ集合中のグループの数だけbitを設け、その評価がグループに関連するものであればtrueを格納しておく.