【ICDE2014勉強会】

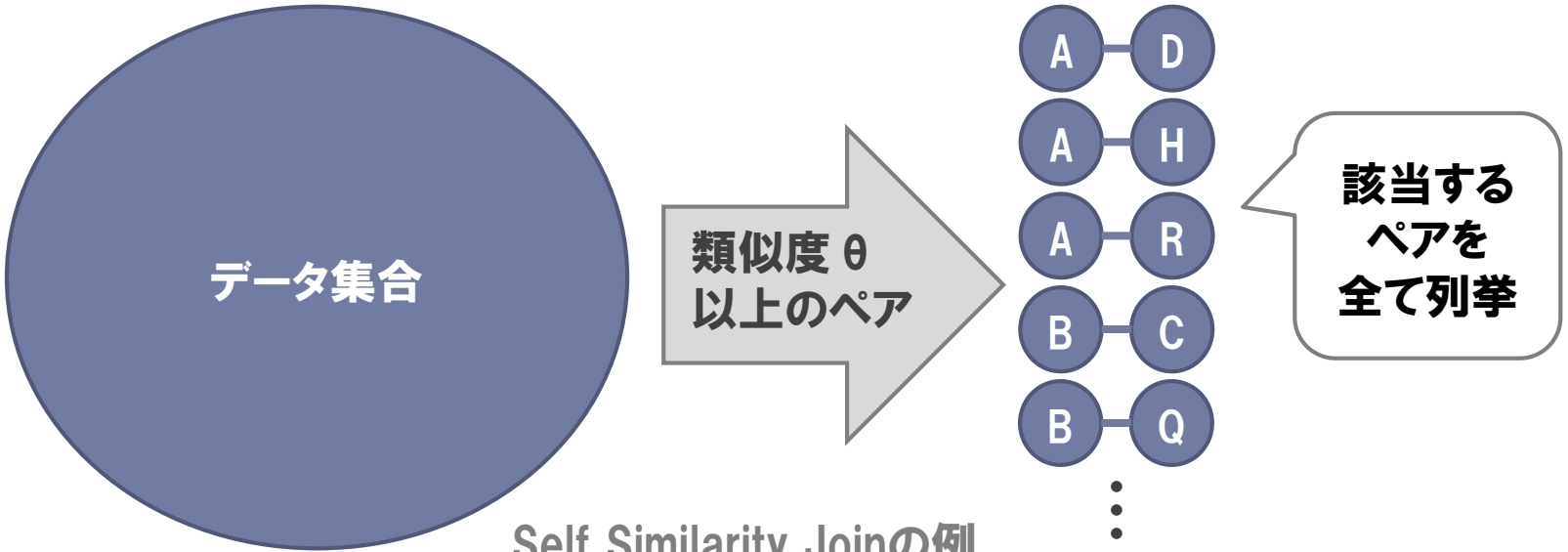
Session 22: Similarity Joins

担当：白川（大阪大学）

まずはじめに…

- Similarity Joinとはなんぞや？
  - 2つのデータ集合の中から類似するデータペアを全部見つけるタスク

- Self Similarity JoinとかSimilarity Self-Joinというのもあるが？
  - 1つのデータ集合の中から類似するデータペアを全て見つけるタスク
  - やってることは似たようなもの



Self Similarity Joinの例

Session 22: Similarity Joins の論文

DE140108.PDF

L2AP: Fast Cosine Similarity Search with Prefix L-2 Norm Bounds

David C. Anastasiu, George Karypis

DE140143.PDF

PHiDJ: Parallel Similarity Self-Join for High-Dimensional Vector Data with MapReduce

Sergej Fries, Brigitte Boden, Grzegorz Stepien, Thomas Seidl

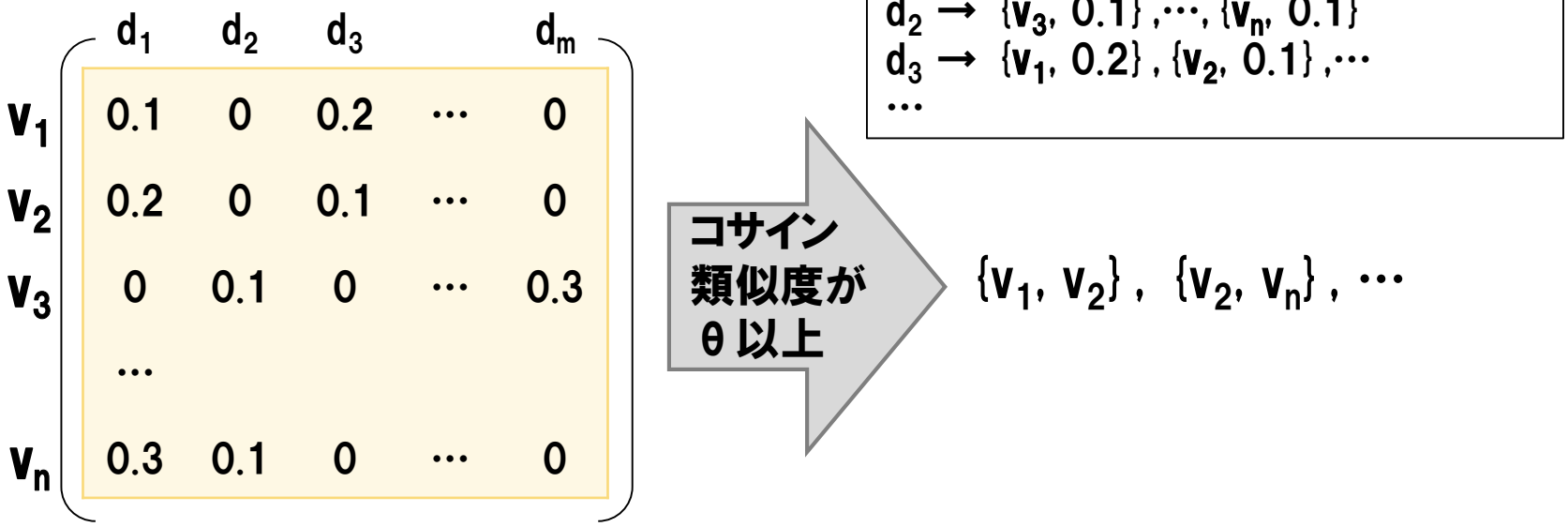
DE140387.PDF

Melody-Join: Efficient Earth Mover's Distance Similarity Joins Using MapReduce

Jin Huang, Rui Zhang, Rajkumar Buyya, Jian Chen

研究背景とか

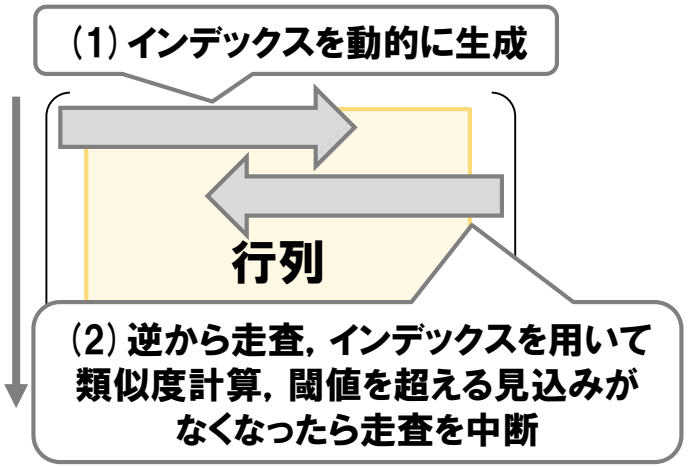
- **超高次元かつ疎なベクトル集合を対象とし，コサイン類似度が閾値θ以上のベクトルペアを全て見つけたい**
  - 計算量が$O(n^2m^2)$，ただしnはベクトル数，mは次元数
  - よくあるやり方：転置索引を作って非零成分にのみアクセス
  - ベクトルが正規化されていればdot(非零成分同士の積)で計算可能



提案手法

*1 ソフトウェア http://www-users.cs.umn.edu/~dragos/l2ap/

- **AllPairs** [Bayardo, WWW07]
  - 動的索引や，見込みのないベクトルペアの早期切り捨てを行う手法
  - MMJoin [Lee, DEXA10]：
    切り捨てをより早期に行う拡張手法を提案
  - 走査中のベクトルの未走査部分を x' として
    $dot(x', y) \leq \frac{1}{2}\|x'\|^2 + \frac{1}{2}$ で類似度を推測



各ベクトル順に (1) (2) の処理

- **提案手法：L2AP*1**
  - コーシー＝シュワルツの不等式を利用して切り捨てをさらに早期に！
    $\mathrm{dot}(\boldsymbol{x}', \boldsymbol{y}) \leq \|\boldsymbol{x}'\| \times \|\boldsymbol{y}\|$ （yは正規化されているのでノルムが1）
  - 類似度の推測値 $dot(x', y) \leq \|x'\|$ は必ず既存手法より正確
  - 証明：$(\|x'\| - 1)^2 \geq 0 \Rightarrow \frac{1}{2}\|x'\|^2 + \frac{1}{2} \geq \|x'\|$ より，
    L2APによる推測値のほうが必ずドットプロダクトの真の値に近い！

評価

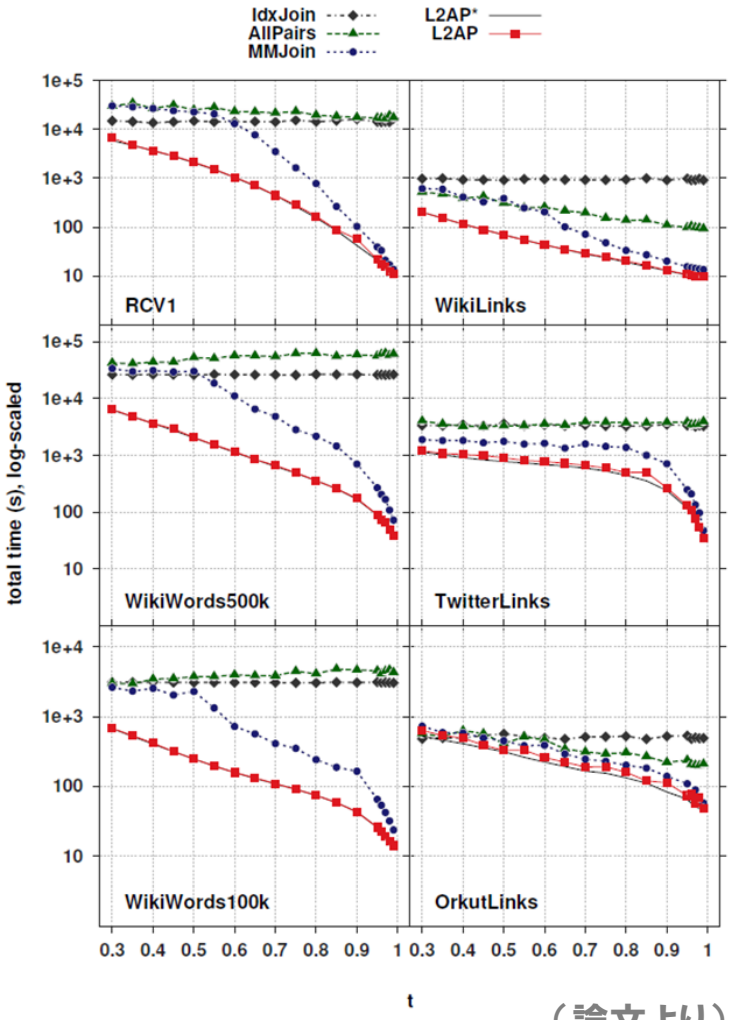
- **6つの実データを使って評価**

  nnz: 行列全体での非零成分数
  mrl: 1行の平均非零成分数
  mcl: 1列の平均非零成分数

Dataset	n	m	nnz	mrl	mcl
RCV1	804414	43001	61e6	76	1417
WikiWords500k	494244	343622	197e6	399	574
WikiWords100k	100528	339944	79e6	787	233
TwitterLinks	146170	143469	200e6	1370	1395
WikiLinks	1815914	1648879	44e6	24	27
OrkutLinks	3072626	3072441	223e6	73	73

（論文より）

- **閾値 t を変えてもたいていの場合L2APが最速！**
  - インデックスサイズや途中結果のデータサイズも減っていたので省スペース！
  - BayesLSH [Satuluri, VLDB12] などの近似手法と比べた場合でも多くの場合L2APが勝っていた！（詳しくは論文参照）



（論文より）

研究背景とか

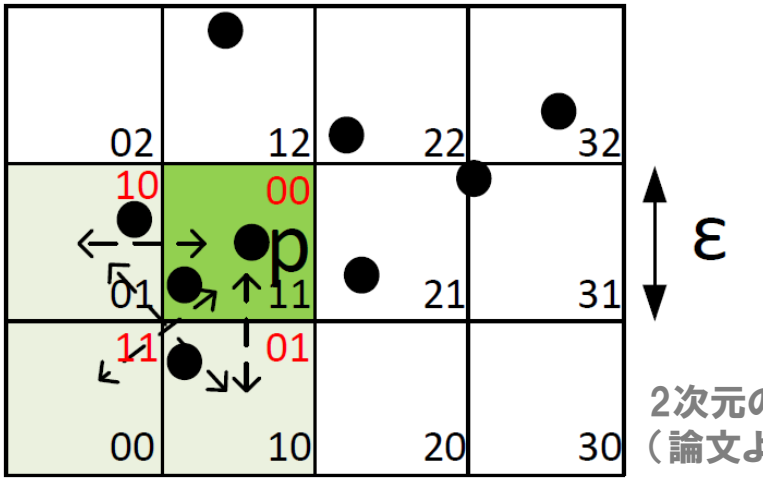
- 高次元なベクトル集合を対象とし，距離が ε 以下のベクトルペアを，MapReduceを使って全て見つけたい

BTW:ドイツのデータベースの会議

- MR-DSJ [Seidl, BTW13]：グリッドベースの手法（同じ研究グループによる成果）
  - d次元のベクトルをd次元のグリッドに配置，Reducerが各セルを担当
  - 近くのグリッドに位置するベクトル間のみ距離を計算
  - 緯度経度で距離計算とかなら問題ないけど，高次元だと隣接セルの数が指数関数的に増加してしまう！

→ 通信コスト増大！

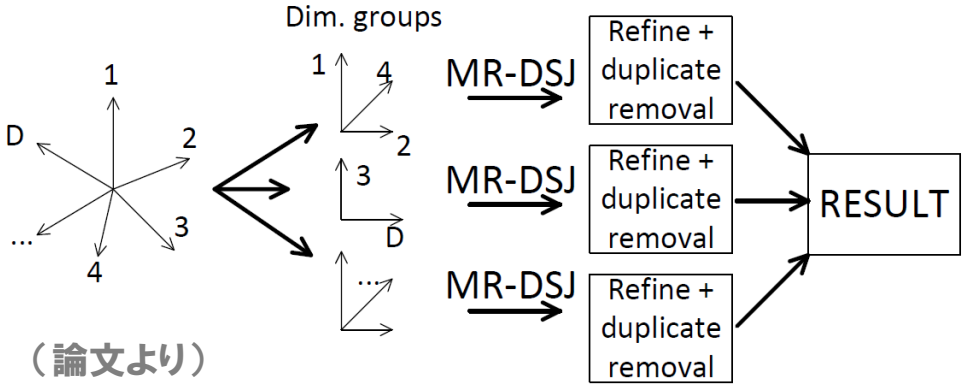
濃い緑のセルを担当するReducerは矢印で示したセル間を担当



2次元の場合（論文より）

提案手法

- 高次元空間をk個の次元グループに分けてMR-DSJを実行
  - 各次元グループでの結果を最後に集約
  - 次元グループ数は次元数に対して線形なのでだいぶマシ！



（論文より）

- 各次元グループに対しては，よりタイトな閾値 $\varepsilon_k = \varepsilon / \sqrt[m]{k}$ を使うことで候補のペアを減らせる

Lm-norm距離の場合

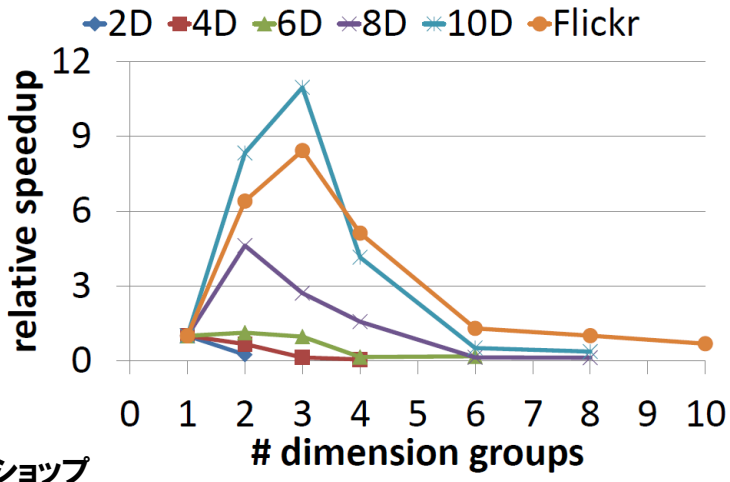
  - Q: そんなことしたら解を見逃すケース(＝偽陰性)も発生するのでは？
    A: ありえない，なぜなら別の次元グループの出力として得られるから

- グリッドが均等だと一部のセルにデータが偏る(MR-DSJでも問題だった)
  - データが少ないところは数が均等になるよう広い範囲を担当させる
    （これは本研究とは独立した問題だから詳細は省く，みたいなこと書いてて簡単な説明しかなかった…）

評価

*1 http://press.liacs.nl/mirflickr

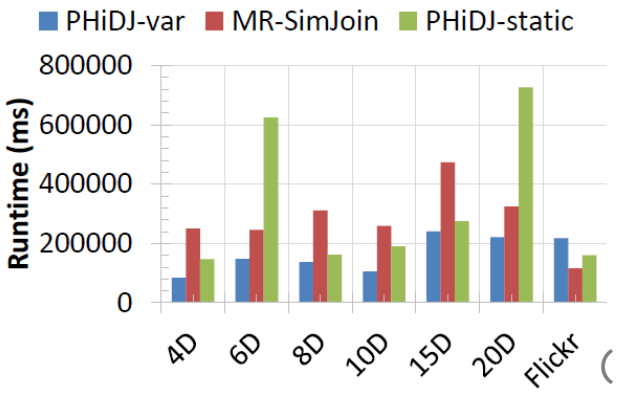
- 2次元～20次元の疑似データ＆10次元のMIRFLICKR*1データ(右図2つ)
  (もっと高次元なデータでの結果が見たかった…)
  - 8次元ぐらいから効果が出る
  - Theta-join [Okcan, SIGMOD11]より高速
    (もっとも，この手法は任意のJoinが可能だが)



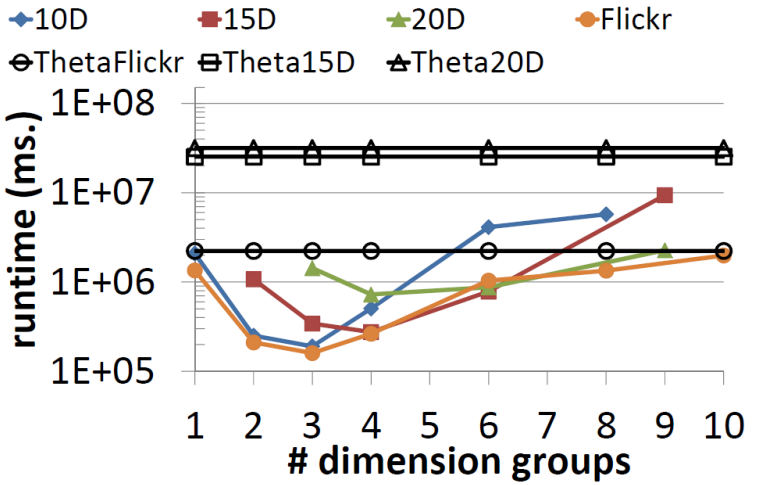
(論文より)

Cloud-I 2012:VLDB 2012のワークショップ

- MR-SimJoin [Silva, Cloud-I12]と比較(左下図)
  - Flickr以外では倍ぐらい速かったが，肝心の実データで負けるのはどうなの…



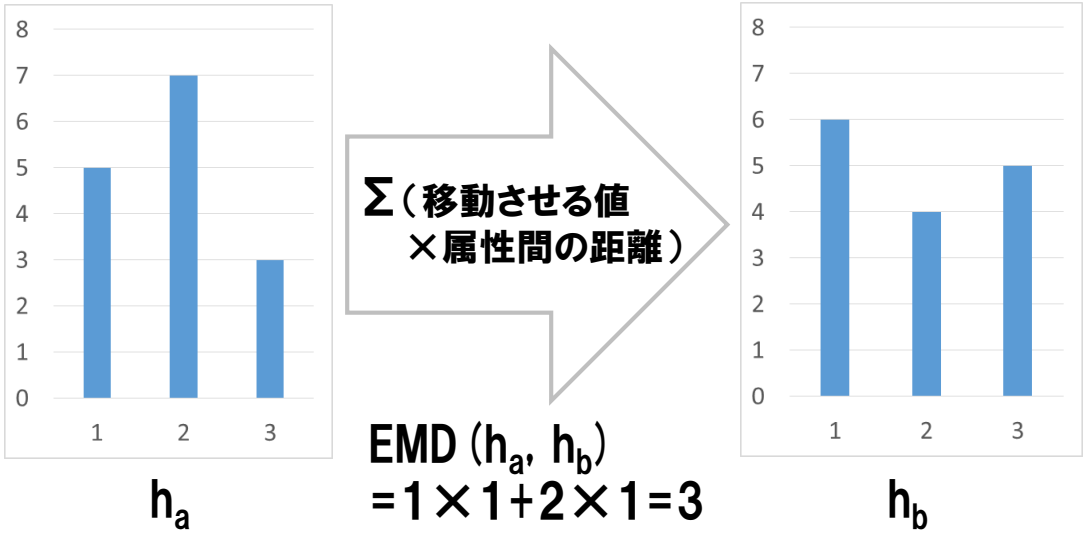
(論文より)

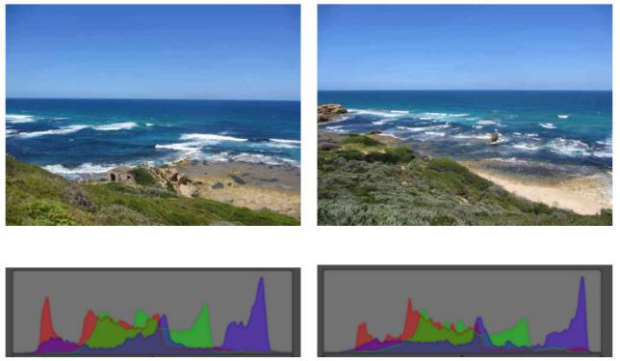


(論文より)

研究背景とか

- 1～3次元程度のヒストグラムの集合を対象とし，EMDが ε 以下のヒストグラムペアを，MapReduceを使って全て見つけたい
  - EMD（Earth Mover's Distance）：一方のヒストグラムから他方のヒストグラムになるまでに要する値の総移動コスト（文字列の編集距離に似てる）





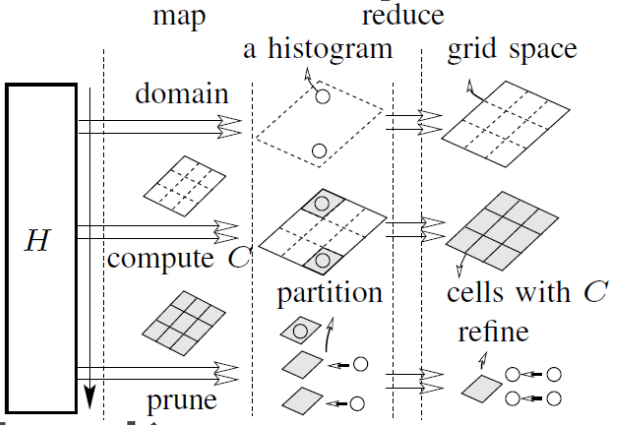
類似画像判別（発表スライドより）

  - 画像検索 [Ruzon, IEEE TPAMI01]，ジェスチャ認識 [Ren, ACM MM11]，重複検出 [Xu, CVPR08]，確率データマイニング [Xu, VLDBJ12] など幅広い応用

提案手法

*1 長さrと角度θの組の集合で表現された空間
http://ja.wikipedia.org/wiki/%E3%83%8F%E3%83%95%E5%A4%89%E6%8F%9B

- **ハフ空間*1に移し[Ruttenberg, VLDB12]，近そうなペアのみ計算**
  これを3回のMapReduceジョブで並列処理させ，Similarity Join化
  - 1回目で各ヒストグラムをハフ空間に写像
  - 2回目でハフ空間の各セルについてLower Bound(LB)に使うエラー値Cを算出
    (実際の累積頻度とNormal Distributionによる回帰の誤差より算出)
  - 3回目で各ヒストグラムごと，各セルごとに，
    A) 自分の位置するセルならネイティブレコード
    B) それ以外のセルでかつLBがε以下ならゲストレコード
    として，ヒストグラムとセルの組をMap
  - 3回目のReducerが各セルを担当，ネイティブ→ネイティブ，ゲストのEMD計算
    ここまでの処理で近そうなペアとして残った



処理の流れ(論文より)

いや説明全然分からないんだけど

とにかく，画像処理とかでよく使われるハフ空間に移すことでEMDが近そうなペアを効率的に見つけられるので，それをMapReduceに対応させた，という感じです

評価

*1 http://archive.ics.uci.edu/ml/datasets/Corel+Image+Features

- **5つの実データセットで評価**

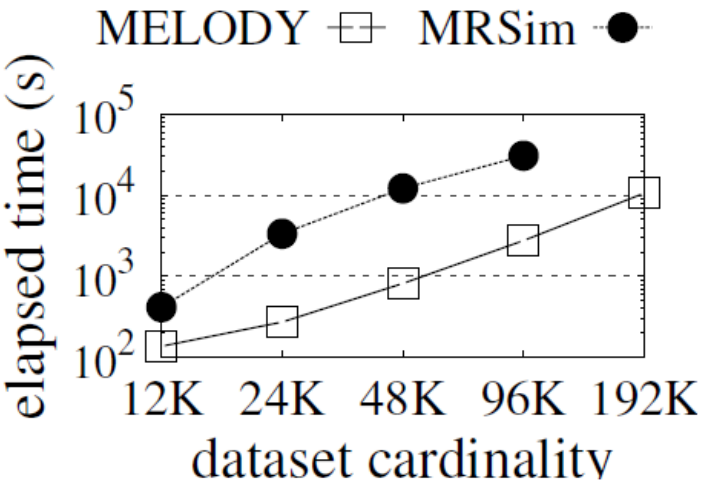
  COREL*1:Color (CC), Layout (CL)
  MIRFLICKR:Vertical (MV), Horizontal (MH), Slash (MS)
  (このスライドではMVの結果のみ表示)

その他セッティング

写像関数:3種類
グリッド:4×4セル
Hadoopインスタンス:48

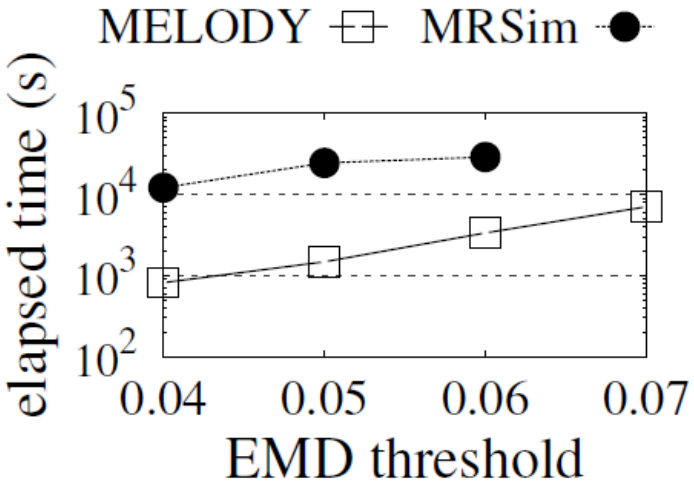
1つ前の論文に出てきた[Silva, Cloud-I12]と同じ手法

- **MR-SimJoin [Silva, SIGMOD12, Demo]と比較**
  - **画像の数(cardinality)やEMDの閾値を変えても提案手法が常勝**



(c) MIRF. Vertical (MV)

(論文より)



(c) MIRF. Vertical (MV)

(論文より)