1G-02

メニーコアプロセッサを用いた集合間類似結合の高速化

菅野健太 † 天笠俊之 ‡ 北川博之 ‡

† 筑波大学情報学群情報科学類 ‡ 筑波大学計算科学研究センター

1 はじめに

集合間類似結合は集合をレコードとする 2 つのテーブルから類似する集合の対を抽出・列挙する処理であり，文書間のデータクリーニングや実体解決などに用いられる．この処理の問題点として，大規模なテーブルに対する処理の計算コストが大きいことが挙げられ，効率的な処理に関する様々な研究が行われている [2]．また，近年高性能な並列計算機としてメニーコアプロセッサが注目されている．そこで，本稿では，メニーコアプロセッサを用いた集合間類似結合の高速化手法を提案する．

2 Intel Xeon Phi

Xeon Phi (Knights Landing) は Intel 社製のメニーコアプロセッサである．その特徴として，(1) 約 70 個の物理コアがそれぞれ四つのハードウェア・スレッドを実行できること，(2) 各コアが 512-bit 幅の SIMD 命令をサポートするユニットを持つこと，(3) 高バンド幅のメモリ (MCDRAM) を持つこと，などが挙げられる．これらの特徴を活かすために，プログラムを適切に設計することが重要である．

3 集合間類似結合

まず，3.1 節で集合間類似結合の定義を述べる．次に，3.2 節で b-bit MinHash に基づく集合間類似結合の説明をする．

Accelerating Set Similarity Join using Many-core Processor
Kenta SUGANO†, Toshiyuki AMAGASA‡ and Hiroyuki KITAGAWA‡
†College of Information Science, University of Tsukuba
‡Center for Computational Sciences, University of Tsukuba

3.1 集合間類似結合とは

集合間類似結合は，集合からなるレコード集合 R, S，集合間類似度関数 $\mathrm{Sim}(\cdot,\cdot)$，閾値 t に対して，$\{\langle r, s\rangle \in R \times S \mid \mathrm{Sim}(r, s) \geq t\}$ を求める処理として定義される．集合間類似度関数としては一般に Jaccard 係数や Cosine 類似度などの様々な類似尺度が用いられる．本稿では，集合間類似度関数として Jaccard 係数を用いる．Jaccard 係数を用いることの利点の一つに，後述する Minhash 法を使った Jaccard 係数の推定アルゴリズムが利用可能な点が挙げられる．なお，集合 r, s 間の Jaccard 係数は，$\mathrm{Jac}(r, s) = |r \cap s|/|r \cup s|$ として定義される．

3.2 b-bit MinHash

Li らが提案した b-bit MinHash [3] は Jaccard 係数の推定値を効率的に計算するための手法である．b-bit MinHash を用いた集合間類似結合はシグネチャ生成と結合処理の 2 段階から構成される．

まず，テーブル中の各集合をシグネチャと呼ばれるデータ構造に変換して，シグネチャをレコードとするテーブルを構築する．集合 $s \subseteq \Omega = \{0, 1, ..., D-1\}$ のシグネチャは，定数 b と k 個のハッシュ関数 $\pi_1, ..., \pi_k : \Omega \mapsto \Omega$ に対して，$\mathrm{Sig}(s; b, \pi_1, ..., \pi_k) = \left\langle \min_{e \in s}\left(\pi_1(e)\right) \,\&\, m_b, ..., \min_{e \in s}\left(\pi_k(e)\right) \,\&\, m_b \right\rangle$ として定義される．m_b は下位 b bit のみが 1 のビット列である．

次に，シグネチャをレコードとする二つのテーブルに対して結合処理を行う．全てのシグネチャの対について，シグネチャの各要素を比較することで得られる Jaccard 係数の推定値が閾値 t を超えるならば，対応する集合の対を出力する．$\mathrm{Jac}(r, s)$ の推定値は，$\hat{\mathrm{Jac}}(r, s) = 1 - \mathrm{Ham}\left(\mathrm{Sig}(r), \mathrm{Sig}(s)\right) / \left\{\left(1 - 2^{-b}\right) k\right\}$ で得られる．なお，$\mathrm{Ham}(\cdot,\cdot)$ はハミング距離を示す．

Copyright ©2017 Information Processing Society of Japan.
All Rights Reserved.

4 提案手法

本稿では，b-bit MinHash に基づく集合間類似結合に対して，Intel 社製のメニーコアプロセッサである Xeon Phi (Knights Landing) による高速化手法を提案する．

4.1 シグネチャ生成

シグネチャの生成において，テーブル中の各集合に対する処理をマルチスレッドによって並列化する．また，シグネチャの単一の要素の計算において，集合の各トークンのハッシュ値の計算を SIMD 演算によって並列化する．生成したシグネチャは Xeon Phi 上の高バンド幅メモリである MCDRAM 上の連続した領域に格納する．

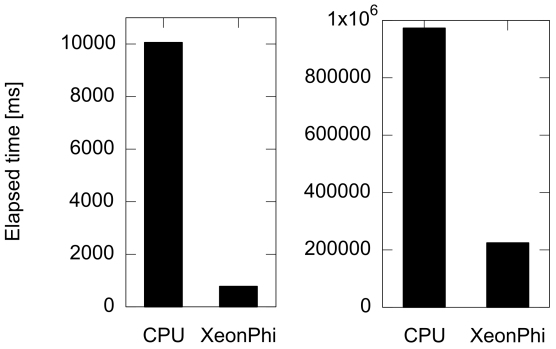
4.2 結合処理

結合処理には，Ambuj らが提案した Blocked Nested Loop 結合 [4] を用いる．これは，Nested Loop 結合において内側ループで参照されるテーブルを連続する α 個のレコードからなるブロックに分割し，外側ループで参照されるテーブルと各ブロックに対して Nested Loop 結合を行う手法である．ブロックがキャッシュに収まる程度にパラメータ α を小さくすることで，結合処理中のキャッシュミスの削減が期待できる．また，それぞれの Nested Loop 結合において，外側のループを展開してマルチスレッドによる並列化を行う．さらに，Jaccard 係数の推定値の計算において，シグネチャの各要素の比較を SIMD 演算によって並列化する．Jaccard 係数の推定値が閾値を超える集合の対は，スレッド毎に割り当てられた出力用領域に格納する．

5 実験

提案手法と CPU 上の b-bit MinHash を用いた集合間類似結合の実行時間の比較を行う．データセットとしては，CiteSeerX [1] から得た複数の論文の Abstract に対して出現した単語からなる集合を生成し，それらをレコードとするテーブルを構築して使用した．テーブル中の集合の数は 606,246，集合のサイズの平均は約 85，全体集合のサイズは 809,371 である．また，各パラメータについては，$t = 0.7$，$b = 1$，$k = 1,024$，$\alpha = 208,896$ とした．

図 1 に実験結果を示す．実験結果より，提案手法は CPU 上の処理と比較して，シグネチャ生成では約 13.0 倍，結合処理では約 4.3 倍，全体では約 4.4 倍の高速化となったことがわかる．



(a) シグネチャ生成　**(b)** 結合処理

図 1: 実行時間の比較

6 まとめ

本稿では，メニーコアプロセッサを用いた集合間類似結合の高速化手法を提案した．今後は手法の改善をする予定である．

7 謝辞

本研究の一部は，科研費・基盤研究 B(#26280037) の助成を受けたものである．

参考文献

[1] CiteSeerX. `http://citeseerx.ist.psu.edu/`.

[2] Dong Deng, Guoliang Li, He Wen, and Jianhua Feng. An efficient partition based method for exact set similarity joins. *PVLDB*, 2015.

[3] Ping Li and Arnd Christian König. b-bit minwise hashing. In *WWW*. ACM, 2010.

[4] Ambuj Shatdal, Chander Kant, and Jeffrey F. Naughton. Cache conscious algorithms for relational query processing. In *VLDB*, 1994.

Copyright ©2017 Information Processing Society of Japan.
All Rights Reserved.