2P-1

クラスタリングを利用したTop-k Join処理

鈴木 貴敦† 高須 淳宏†† 安達 淳††
†東京大学大学院 ††国立情報学研究所

1 はじめに

Complex Search Task とは，複雑なクエリによる高度な検索を実現することを目標とした検索タスクである [1]．ここでの複雑なクエリとは，例えば，“どこに行けば自分の専門分野の学会に参加しつつ，浜辺で休養もできるか” のように，回答するためには複数のデータベースを横断的に調べる必要のあるもののことを言う．我々は Complex Search Task のなかでも特に Top-k Rank Join に関する問題について取り組んでいる．Complex Search Task における Top-k Rank Join の問題とは，クエリ Q が複数のサブクエリ $q_1, q_2, \ldots, q_m$ に分解可能であり，クエリへの回答はサブクエリの回答の組み合わせで表されるとした場合に，各サブクエリの回答リストから最大で 1 つずつオブジェクトを選んで組み合わせを作り，スコアの高い上位 k 件を求めることである．

Complex Search Task における Rank Join は，これまでのように単一なデータベースに関して検索するのではなく，対象が複数のデータベースにまたがっているため，必ずしも Join 可能でないことが問題となる．先ほどの例で言えば，東京で行われる学会に対しては，沖縄のビーチと一緒に提示できないということである．そのため，これまでの Top-k Join の手法をそのまま適用することはできない．それに対して Sharem らは，Join が可能でないことを考慮した Top-k Join の手法を提案した [2]．この手法は，Nested Loop と呼ばれる組み合わせを求めるための基本的な手法に，Top-k Join で用いられる手法を応用したものである．既存手法の問題点は，すべてのサブクエリから得られる回答リストを一括して扱わなければならない点である．[3] で指摘されているように，Top-k Join では，各リストを非同期的に処理することで処理効率が上がる可能性がある．そこで我々は，サブクエリの回答リストをクラスタリングし，各クラスタについて部分的に Join した後，全体の解を求める手法を提案した．

Using Clustering in Top-k Join Queries
† Takanobu Suzuki(suzuki@nii.ac.jp)
†† Atsuhiro Takasu(takasu@nii.ac.jp)
†† Jun Adachi(adachi@nii.ac.jp)
Graduate School of Information Scienece and Technology, The University of Tokyo (†)
National Institute of Informatics (††)
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan

2 問題定義

クエリ Q が，サブクエリ $(q_1, q_2, \ldots, q_n)$ に分割可能であるとする．サブクエリ $(q_1, q_2, \ldots, q_n)$ の回答リスト L を $\mathrm{L} = (L_1, L_2, \ldots, L_n)$ とする．各 $L_i = (o_{i1}, o_{i2}, \ldots, o_{im_i})$ 中のオブジェクトにはそれぞれ得点が割り当てられており，正規化されているとする．オブジェクトの組み合わせ g は，各リスト $L_1, \ldots, L_n$ からそれぞれ最大で 1 つオブジェクトを選び作られる組み合わせであり，任意の 2 つのオブジェクトが Join 可能であることが組み合わせ成立の条件とする．任意の 2 つのオブジェクトの Join 可能性を判別する関数を $jcon$ とする．組み合わせの得点は，重み付けベクトル $\mathrm{w} = (w_1, w_2, \ldots, w_n)$ が与えられるため，各オブジェクトの重み付き和で表す．

これらの前提のもと，本稿で扱う問題は，L, k, $jcon$, 各リストの重み w が与えられたときに，最も得点の高い k 組のオブジェクトの組み合わせ列 $G = (g_1, g_2, \ldots, g_k)$ を求めることである．

3 提案手法

提案手法は，既存の VNL，HNL，VHNL にクラスタリングを適用した CVNL，CHNL，CVHNL の 3 つである [2]．効率よく枝刈りを行うには，クラスタから得られる組み合わせの数を小さくすればよいので，Join が成功する確率が低いもの同士でクラスタを構成すればよいが，今回は，クエリが決まって初めて処理対象のサブクエリの回答リストが得られるので，クラスタリングに時間を割くことは好ましくない．また，同期的に全ての回答リストが得られるとは限らないことも鑑みて，今回は，リストのペア，もしくはトリプレットを無作為に生成し，処理を行うこととする．

3.1 CVNL，CHNL

CVNL(Cluster VNL) では，まず始めに，サブクエリに対する回答リストのクラスタを作成した後，各クラスタについて，VNL を利用して可能な組み合わせを全て生成し，得点の降順に整列した部分解のリストを作成する．各部分解のリストが得られたら，それらについて再度 Join を行うことで解を求める．

CHNL(Cluster HNL) では，CVNL と同様に，まず始めにサブクエリに対する回答リストのクラスタを作成する．その後，各クラスタから 1 組以上 Join が成立するまで HNL を行う．各クラスタから Join 結果が得られたら，これまでに得られた Join 結果と再度 Join を

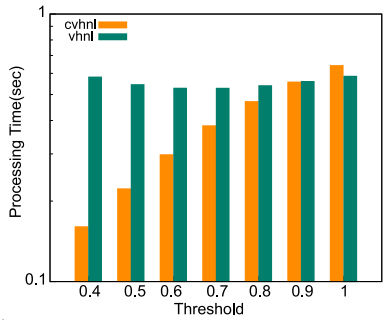

図 1: CMU Face Images

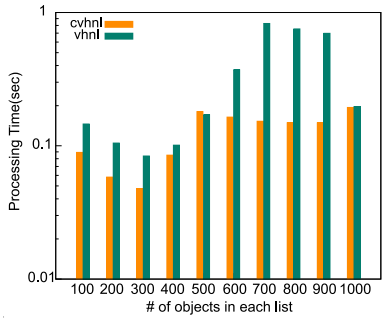

図 2: 人工データ (サブクエリ数 4)

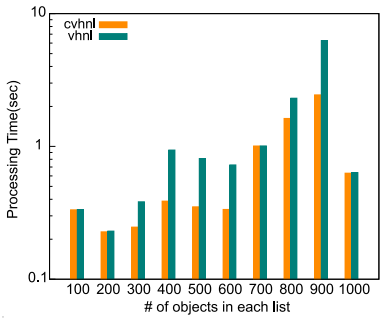

図 3: 人工データ (サブクエリ数 6)

行い，解を求める．解が k 組得られたら，k 番目に得点の大きい Join 結果 g_k と，今後得られる可能性のある Join のうちで，最も得点の高いものと比較を行い，g_k のほうが大きければ処理を終了する．

CVNL の利点は，CHNL よりも早い段階でオブジェクトを選ばない選択肢が出現する点である．そのため，オブジェクトの Join が成立しにくいときに有利な手法であると言える．ただし，必ずしも得点の大きい組み合わせから順に生成するわけではないため，オブジェクトの Join が成立しやすい場合に，余分な組み合わせを生成してしまうので，処理効率が低下する．CHNL の利点は，リストの先頭から順に Join を行うため，得点の高い組み合わせが優先的に出現するため Join が成立しやすい場合に効率よく処理が行える点である．しかし，Join が成立しにくい場合，オブジェクトを選ばないという選択肢は探索の最後に出現するため，処理効率が低下してしまう欠点がある．

3.2 CVHNL

CVHNL は，CVNL と CHNL を合わせた手法である．Join が成功するオブジェクトペア数と，すべてのオブジェクトペア数から求められる Join が成功する確率を用いて，リストの先頭からいくつ調べれば k 個の組み合わせが得られるかの期待値を求める．その期待値以前では CVNL を行い，以降は CHNL を行う手法である．閾値の求め方は [2] を参照されたい．

4 評価実験

実験には，CMU Face Images(サブクエリ数 4 個，各サブクエリにつき 420 個のオブジェクト) を用いて生成したデータと，正規乱数を用いて生成した人工データ (サブクエリ数 4 個，6 個．それぞれについて Join 成功確率 10%) を利用した．提案手法，既存手法それぞれに対し，上位 1 組を求める場合について，100 回実験を行い，平均計算時間を比較した．CMU Face Images に関しては，クエリとして，“複数の異なるカメラに写る人物のうち，どのカメラにおいても挙動が特異な人物はどれか” を想定している．横軸は，同一人物判定を行う際の閾値であり，小さいほど厳しく判定を行うため，Join が成功する確率は小さくなる．CMU Face Images は，UCI Machine Learning Repository で提供されているデータセットである．

図 1,2,3 にそれぞれ CMU Face Images，サブクエリ数 4 個の人工データ，サブクエリ数 6 個の人工データから得られた結果を示す．図より，提案手法は，サブクエリ数が小さいとき，または Join が成功する確率が小さい場合は有利であるが，サブクエリ数が増加したり，Join が成功する確率が大きくなるにしたがって，性能が落ちてしまうことがわかる．そのため，今後の課題としては，サブクエリ数の増加と Join が成功する確率が大きい場合への対応が挙げられる．

5 おわりに

本稿では，クラスタリングを用いた Top-k Join 処理の高速化手法を提案した．提案手法では，サブクエリの回答リストをクラスタリングすることで，各リストについて非同期に処理を行い，より効率よく処理がすすめることができた．ただし，サブクエリ数が増加したり，Join が成功する確率が高い場合に，性能が落ちてしまうため，今後は，そのような条件に対しても効率よく動作する手法を検討する．

また，今回，提案手法は非同期の手法であるが，並列処理が行えていない．今後，非同期の強みを活かすため，並列化を行い，さらなる処理速度の向上を目指す．

参考文献

[1] S. Ceri and M.Brambilla, Search Computing Challenges and Directions, Springer LNCS, 2010.

[2] Sharem, et al., Computing the Top-k Maximal Answers in a Join of Ranked Lists, CIKM, 2010.

[3] C. Wu, et al., Efficient Parallel Top-k Computation Algorithm using Symmetry Breaking, IPSA, 2010