5M-1

空間データベースにおけるm-最近接キーワード検索の一方式

邱 原† 大森 匡† 新谷 隆彦† 藤田 秀之†
†電気通信大学大学院情報システム研究科

1 はじめに

近年, 位置情報を持つ Web 上のデータ (twitter や Flickr など) からの情報抽出・検索が研究されている. Zhang らが提出した mCK(m-Closest Keywords) 検索はこの方式の一種である [1].

mCK 検索とは m 個のキーワードの入力を受けて, 当該キーワードを満たす m 個のオブジェクトで構成されたオブジェクト集合 T のうち, 最も m-オブジェクトが近接している集合 T_{opt} を探す問題である. T の近さは「直径」(T の中の任意 2 オブジェクトの間の距離の最大値) によって決める. T の直径が小さければ小さいほど全オブジェクトが近いと言える.

この問題の本質は, 探索空間を各キーワードに関連するオブジェクト集合の組み合わせとした時, 最小直径を持つオブジェクト集合を探すことである. 本稿は mCK 検索について「早めにより近いオブジェクト集合を探し, 探索空間の走査を減少する」方法として,Diameter Candidate Check(DCC) 方式を提案する.

2 mCK 問題

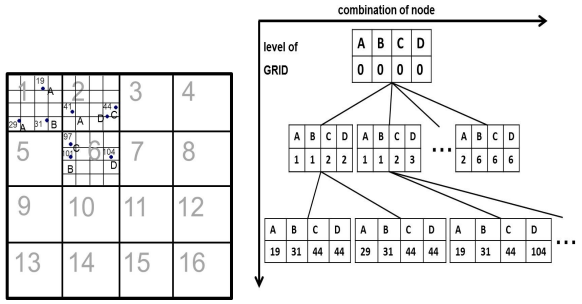
2.1 前提

先行研究として, 全データを 1 つの R-tree で保存することを前提に,Apriori で MBR の組を列挙する手法 [1] がある. しかし, 現実的には,twitter や Flickr などのサーバ側では Grid 分割した階層データ構造によるデータ管理しか期待できない. そこで我々は, 階層グリッド分割木でデータ管理された場合に,Nested loop でノードを組合せしながら深さ優先で探索空間を走査し, mCK に適応した枝刈り戦略を入れた探索方法 (nested loop with sophisticated pruning, NLwP) を [2] で述べた.

本稿は Grid に基づいてセルあたりキーワードの MBR をつけるデータ構造を前提として展開する.

2.2 nested loop with sophisticated pruning 法

NLwP のデータ分布の例と探索空間木を図 1 に示す. $\{A, B, C, D\}$ はキーワードであり, 数字は Grid 上のノード番号を表す. 記号 $A[i]$ はキーワード A に関連する番号 i のノードを示し, 各キーワードに関連するノードの組み合わせ $\{A[i], B[j], C[k], D[l]\}(i, j, k, l$ はノード番号である) はノードセットとする. 図 1 で枝刈り戦略は探索空間木の深さ優先探索によって行う. この方法は,mCK に応じた枝刈り戦略を持つが, ノードの組み合わせ (ノードセット) を深さ優先で展開するため枝刈り効率がまだ低い.



(a) データの分布 (b) 探索空間木

図 1: mCK 検索の例

3 Diameter Candidate Check

3.1 DCC の概要

上記のを改良するため,NLwP に対して, さらにノードセットの展開する場合, 全ての子ノードセットを生成し, そのうちより直径が小さい子ノードセットを優先して選ぶという考えがある. しかし, そのままでは子ノードセットの数が指数的に増加する.

そのため Diameter Candidate Check(DCC) 方式を提案する. オブジェクト集合の近さは, 任意の 2 オブジェクト間の距離の最大値 (直径) によって決まる. つまりオブジェクト集合は全部の m 個のオブジェクトを使用せず, 二つのオブジェクトだけ用いても直径の候補は判る. そこで, 二つのオブジェクトから構成されるペアの集合を作って, 小さい順で各ペアにおいて直径としたオブジェクト集合を組み立てることができるかどうかを判定する方が効率がよい.

図 1 の例とした DCC の流れは図 2 によって示す. ここでノードのペアは Diameter Candidate と呼ぶ.

3.2 Candidate Enumeration

Candidate Enumeration は Diameter Candidate の集合を作ることである. 入力としたサイズ m のノードセットの中で 2 ノードを 1 組として選び, 子ノードを展開して Diameter Candidate とする. 例:

A new Algorithm for m-Closest Keyword search on spatial data
† Y.Qiu,T.Ohmori,T.Shintani,H.Fujita
The University of Electro-Communications(†)

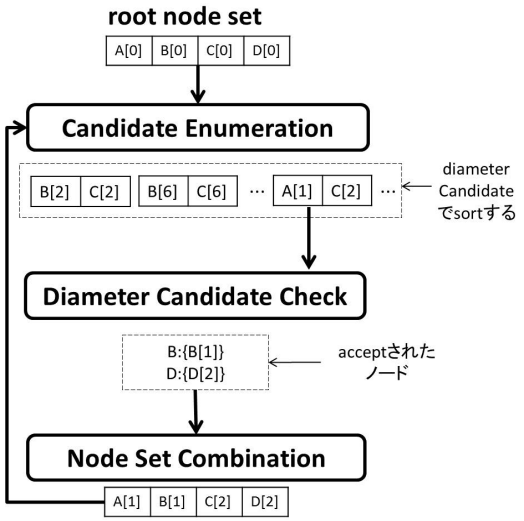


図 2: DCC の流れ

図 2 入力のノードセット $\{A[0], B[0], C[0], D[0]\}$ について,$\{\{A[0], B[0]\}, \{A[0], C[0]\}, ..., \{C[0], D[0]\}\}$ 計 6 組のうち, $\{A[0], B[0]\}$ の子ノードを展開して,$\{< A[1], B[1] >, < A[1], B[6] >, < A[2], B[1] >, < A[2], B[6] >\}$ 計 4 Diameter Candidate とする. そして全ての組の Diameter Candidate をノード間の最大距離によってソーティングする.

もしある Diameter Candidate のノード間の最小距離が閾値より小さい場合, 枝刈りできると判定する.

3.3 Diameter Candidate Check

Diameter Candidate Check は,1 つの Diameter Candidate に対して, 各ノードが Diameter Candidate の付近にあるかどうかを判定するということである. 例えば,Diameter Candidate$< A[1], C[2] >$ に対してノード $B[1]$ をチェックする場合

$maxdistance(A[1], B[1]) < maxdistance(A[1], C[2])$
$maxdistance(C[2], B[1]) < maxdistance(A[1], C[2])$

の 2 つ満足すれば,$B[1]$ を accept する.

もしあるキーワードがノードのうち accept されたものがなければ,Diameter Candidate は直径になれないので枝刈りされる.

3.4 Node Set Combination

上記で accept 判定された各キーワードのノードを組み合わせしながら, ノード間の最大距離と Diameter Candidate の最大距離と比較によって枝刈りを行う. 組み合わせたノードセットと Diameter Candidate のノードを合わせたノードセットを生成して, 再帰的に Candidate Enumeration を行う. 例: 図 2 により Diameter Candidate$< A[1], C[2] >$ において,accept されたノードは $B[1]$ と $D[2]$ である. $maxdistance(B[1], D[2]) < maxdistance(A[1], C[2])$ を満たしたら,$B[1], D[2]$ と $A[1], C[2]$ を合わせて, ノードセット $\{A[1], B[1], C[2], D[2]\}$ を生成する.

もし生成されたノードセットが葉ノードのみからなるなら, ノードにあるオブジェクトを抽出して同じ流れ DCC を行なって, 最小直径のオブジェクト集合を求める.

4 実験

実験データとして,keyword 数を 100, 各 keyword に 500 個のレコードを関連づけ, 座標を生成して木に入れる. keyword あたりのレコードの座標の生成方法はランダムで中心点を選んで, 中心点との距離を正規分布で生成する. 標準偏差は一辺の 1/4 の長さである. 入力 keyword の数（m） vs. 実行時間 (Java,Intel Xeon 2.67GHz,10 回平均) を図 3 に示す.

実験結果からみると,DCC は NLwP 法より探索効率が優れるとわかった. 各キーワードに関連するデータは一定程度で密集して, 関連するノードの数が減少できる. また,NLwP 法は最初に良い閾値を求められないが,DCC は小さい直径の候補を優先的に選び, 枝刈り能力が高い.

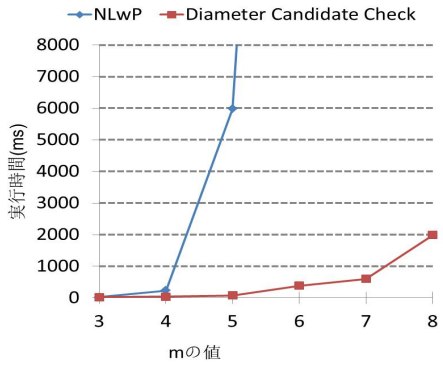


図 3: 入力 keyword の数（m） vs. 実行時間

5 おわりに

本稿では mCK 問題における, 探索順を priority 順に変えることを解決するために,Diameter Candidate Check 方法を提案した. 実行処理効率について,NLwP 法より大幅に改善できるとわかった.

参考文献

[1] D.X.Zhang et al, "Keyword Search in Spatial Databases: Towards Searching by Document," IEEE ICDE, pp.688-699, 2009.

[2] 邱, 他, "空間データにおける 2^n 分割木を用いた m-最近傍キーワード検索" DEIM 2013 A9-5.