データベースシステム 125−50
（2001. 7. 18）

近傍連鎖点列探索における静的索引

佐藤 康裕† 田中 覚‡ 遠山 元道§

†慶應義塾大学大学院 理工学研究科 開放環境科学専攻
‡TIS 株式会社
§慶應義塾大学 理工学部 情報工学科/JST さきがけ研究 21
E-mail: † yas@db.ics.keio.ac.jp, ‡ tana@mmm-keio.net, § toyama@ics.keio.ac.jp

ユーザに検索結果を提示する際、ある基準に従って分類されていると便利である。しかし、ユーザが必要な特徴量を正確に把握することは難しいため、システム側でユーザが必要な情報を得やすくなるような仕組みが必要である。近傍連鎖点列は、ユーザに複雑なパラメータを要求することなく、ユーザの必要とする情報へナビゲートすることを可能にするシステムである。本稿では、近傍連鎖点列の静的な構築方法について提案し、静的索引時の性質と検索、更新方法について述べる。

キーワード：近傍探索 、 類似検索 、 多次元空間 、 索引技術

Static Indexing on Searching in Chained Neighborhood Points

Yasuhiro Sato † Satoru Tanaka ‡ Motomichi Toyama§

†School of Science for OPEN and Environmental Systems,
Faculty of Science and Technology, Keio University.
‡TIS Inc.
§Department of Information and Computer Science, Faculty of Science and Technology,
Keio University. PRESTO,JST
E-mail : †yas@db.ics.keio.ac.jp ‡tana@mmm-keio.net §toyama@ics.keio.ac.jp

It is convenient if it is classified according to a certain standard in case a reference result is shown to users.But, since it is difficult for users to grasp the required amount of the features correctly, the structure which users tend to acquire required information and become by the system side is required. Chained Neighborhood Points make it possible to navigate users to their required information without requiring any complex parameter for them. In this paper, we propose the static construction method of the Chained Neighborhood Points, and describe the reference and the updating method in case of static indexing.

keyword : nearest neighbor search , similarity retrieval , multi-dimensional space , indexing technique

1 はじめに

検索システムが抱える問題のひとつとして、検索結果の分類が挙げられる。検索結果の分類はある基準に従って行われるが、その基準が明確でない場合がある。類似検索がその典型例であり、利用者が検索システムに与える質問が多次元空間の1点に相当する場合、その質問点からの近傍の点の集合が検索結果として返されるケースが多い。ここで問題となるのは、検索結果全てが必ずしも利用者の要望するものではないということである。利用者は複数の検索キーをシステムに与えるが、どの検索キーをどの程度重要視するのかを数値で与えない場合、システム側で利用者の要望を配慮して検索結果を分類し提示する仕組みを備える必要がある。

本稿では「近傍連鎖点列集合」[1] を静的に構築する方法を提案し、静的索引時の近傍連鎖点列集合の性質とそれを用いた検索方法、さらに静的索引時の検索及び更新の方法を示す。この静的索引により、これまでの動的な構築時より検索の効率を良くすることが可能となり、ユーザビリティの向上につながる。

本稿では、2章で最近傍探索アルゴリズムについて述べ、3章で近傍連鎖点列の基本的な性質について述べる。4章で、静的索引時の近傍連鎖点列集合の性質とその検索、更新方法について述べ、5章で検索結果の可視化方法の改善について述べる。最後に6章で結論を述べる。

2 最近傍問合せ

類似検索では、最近傍問合わせという方法が用いられる。最近傍問合せ (Nearest Neighbor Queries[5]) とは、質問点から最も近いオブジェクトを見つける問題である。この最近傍探索アルゴリズムを一般化し、質問点から最も近い上位 k 個のオブジェクトを見つけるように拡張したものが、k-NN(k Nearest Neighbor) 探索アルゴリズムである。対象オブジェクトを R-tree[2] のような索引構造木に格納し、その索引構造木に対して検索が行われる。

近傍検索の順序づけの尺度として次の2つが定義されている [5]。1つ目は MINDIST であり、これは質問点 Q からオブジェクト O までの最小距離 (minimum distance) である。2つ目は MINMAXDIST であり、これは質問点 Q から、オブジェクト O を含む MBR の面 (または辺) までのミニマックス距離 (minimum of the max distance) である。これらの尺度を用いたアルゴリズムにより、R-tree 等を用いた近傍探索の際に不必要な MBR の探索を避け (枝刈り)、検索効率を高めることができる。

[1] Chained Neighborhood Points set (CNP set)

3 近傍連鎖点列を用いた類似検索

近傍連鎖点列を用いた類似検索システムでは、従来の検索のアプローチとは異なっており、利用者が与える検索キーの近傍に存在するオブジェクト集合の集め方に特徴がある。このシステムでは、あらかじめ前処理として近傍探索の際に利用する索引木 (R*-tree[4]) を構築しておく。この R*-tree は、各オブジェクトに検索対象のオブジェクト ID を格納しておき、検索によって得られたオブジェクト ID を用いて DBMS へ問合せを行い、最終的な検索結果を得る。この R*-tree は、検索対象となるオブジェクト集合を多次元空間 (n 次元空間) 内の点にマッピングすることで得られる n 次元ベクトルを元に構築される。この R*-tree を利用することで検索効率を高めている。

3.1 近傍連鎖点列と近傍連鎖点列集合

検索対象となるオブジェクト集合を S、S 内のオブジェクトを $o_i \in S$ で表す。各オブジェクトを n 次元空間内の点で表し、S 内の任意の2オブジェクト o_i,o_j 間の距離を d_{ij} で表すとき、近傍連鎖点列 (CNP) を次のように記述する。

$$CNP = \{o_1 \xrightarrow{d_{1,2}} o_2 \xrightarrow{d_{2,3}} \ldots \longrightarrow o_{d-1} \xrightarrow{d_{(d-1),d}} o_d\} \quad (1)$$

(但し、$d_{1,2} > d_{2,3} > \ldots > d_{(d-1),d}$)

右向きの矢印 ($\rightarrow$) はオブジェクト間のリンク関係を表す。例えば $o_i \xrightarrow{d_{ij}} o_j$ は2点間の距離が d_{ij} であり、o_i から o_j へのリンクが存在することを示している。o_1 を CNP の起点オブジェクト、o_d を終点オブジェクトと呼ぶ。また CNP を構成するオブジェクトの数を CNP の深さとする。距離の記述を省略した場合は CNP を以下のように略記することができる。

$$CNP = \{o_1, o_2, \ldots, o_{d-1}, o_d\} \quad (2)$$

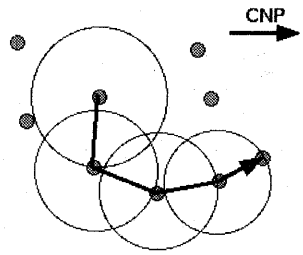

図 1: CNP 基本形

3.2 CNP 基本形

ここで CNP 基本形の構築方法を述べる。はじめに CNP の要素を空 ($CNP = \{\}$) とする。任意のオブジェクト $o_i \in S$ を選択し、o_i を起点オブジェクトとして CNP の要素に加える ($CNP = \{o_i\}$)。次に o_i の最近傍点 o_j を求め、o_j も CNP の要素に加える。このとき o_i,o_j 間の距離を d_{ij} とし、この距離を保持しておく ($CNP = \{o_i \xrightarrow{d_{ij}} o_j\}$)。次に o_j の最近傍点 o_k を求める。このとき $d_{ij} > d_{jk}$ を満たすとき、o_k を CNP の要素に加える ($CNP = \{o_i \xrightarrow{d_{ij}} o_j \xrightarrow{d_{jk}} o_k\}$)。満たさないときは、$o_k$ を CNP の要素に加えず、この時点で 探索を打ち切る CNP に加わった o_k に対しては再帰的に最近傍探索をおこなう。**CNP** は、このような「後戻りすることなく最近傍点を連結して作られるオブジェクトの列」(CNP 基本形) であり、結果的に「隣り合う 2 つのオブジェクト間の距離が単調に減少してする」性質を持つ。

$\forall o_i \in S$ に対して、oi を起点オブジェクトとする CNP を唯一求めることができる。このようにして求めたすべての CNP の集合を総称して近傍連鎖点列集合 (CNP 集合) と呼び、以下のように記述する。

$$CNPset = \{CNP_1, CNP_2, \ldots, CNP_N\} \quad (3)$$

3.3 CNP 拡張形

CNP 基本形では、最近傍オブジェクトのみの探索により CNP を構築した (最近傍点の連鎖)。ここでは任意の $o_i \in S$ を起点オブジェクトとする CNP を構築する際に、o_i の最近傍点上位 s 個に対して再帰的に近傍探索を行うように CNP を拡張する (S 分木的アプローチ)。この拡張により任意のオブジェクトを起点とした CNP が複数得られる。

CNP 拡張形の構築手順は以下の通りである。任意のオブジェクト $o_i \in S$ を選択し、o_i の最近傍点上位 s 個 ($o_{11}, o_{12}, \ldots, o_{1s} \in O_1$) を求める。この時点の CNP 集合は次のようになる。

$$CNPset = \{\{o_i, o_{11}\}, \{o_i, o_{12}\}, \ldots, \{o_i, o_{1s}\}\}$$

各 $o_{1i} \in O_1$ に対して、o_{1i} の最近傍点上位 s 個 ($o_{21}, o_{22}, \ldots, o_{2s} \in O_2$) を求める。CNP の記述式 (1) の距離の制約条件を満たすならば、探索の結果見つかったオブジェクト $o_{2j} \in O_2$ を構築中の CNP の要素に加える。満たさないならば、o_{2j} を CNP に加えず、この時点で構築中の CNP の探索を打ち切る。このような近傍探索を再帰的におこなうことにより個々の CNP を構築していく。

CNP 拡張形では、任意のオブジェクト o_i を起点として、CNP の距離の制約条件 (1) の範囲内で s 分木的に複数のリンクが作成されるため、複数の CNP が構築される (図 2)。

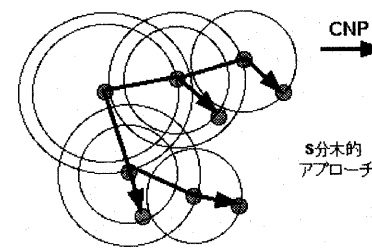

図 2: CNP 拡張形

3.4 CNP 集合を用いた検索

本システム上で類似検索を行うために、n 次元空間上に質問点 ($queryPoint$) を与える。この質問点の最近傍上位 k 個を求め、それらを起点オブジェクトとする CNP 集合を構築する。この集合を構成するオブジェクト群が検索結果の候補となる。実際の検索結果は、質問点から起点オブジェクトまでの距離が、その CNP の最初の距離 (CNP の記述式 (1) の $d_{1,2}$) よりも大きい CNP となる。

検索により発見されるオブジェクトのイメージは図 3 のようになる。

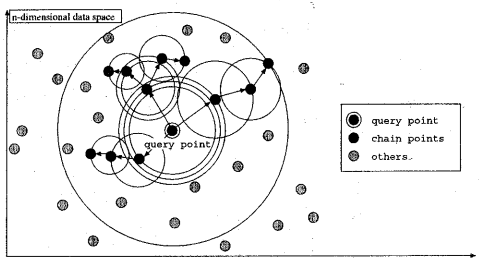

図 3: CNP 集合を用いた検索結果のイメージ

4 CNP集合の静的索引

本システムでは、ユーザから質問が実行されてから質問点に関する CNP 集合を動的に作成している。動的に CNP 集合を作成する方法では、オブジェクトの挿入、削除に関しては、オブジェクトが格納されている索引構造木の更新だけで済ませることができる。しかし、検索が実行されるたびに CNP 集合を作成するために計算しなくてはならないため、ユーザへの応答時間に大きく時間がかかってしまう。また、別の質問で計算した CNP を重複して計算する必要が生じるため無駄なコストも増える。そのため、あらかじめ静的に CNP 集合を作成しておくことで、検索時のユーザへの応答時間を減らすことは有効な手段である。ただし、動的な方法と違い、オブジェクトの挿入、削除のコストが増大するため、更新には工夫が必要となる。

4.1 静的な CNP 集合の性質

静的な CNP 集合には以下のような性質が挙げられる。

CNP 基本型の場合

- 任意のオブジェクトに対して一意的に決まる。
- 複数のオブジェクトから参照されるオブジェクトの CNP は、そのオブジェクトを参照しているオブジェクトの CNP の部分集合となる。
- 任意の CNP の終端オブジェクトからの CNP の長さは必ず 1 である。

CNP 拡張型の場合

- 任意のオブジェクトに対し、k に関して一意的に決まる。
- 任意のオブジェクトからの CNP は最大で k^n 個である。(n:CNP の最大長)
- 任意のオブジェクトを参照する CNP に関して、そのオブジェクトから先の CNP はそのオブジェクトからの CNP 集合の部分集合となる。
- 任意の CNP の終端オブジェクトからの CNP の長さは必ずしも 1 とはならない。

特に CNP 基本型の場合の 2 つ目の性質は重要である。任意の CNP を起点オブジェクトとそのオブジェクトの最近傍のオブジェクトからの CNP で表現することができるため、更新をする際は変更のあったオブジェクトの CNP を変更するだけで、そのオブジェクトを参照しているオブジェクトの CNP も更新できることになる。CNP 拡張型の場合は、式 1 の条件式があるため、起点オブジェクトの最近傍上位 k 個のオブジェクトの CNP 集合全てが必ずしも起点オブジェクトの CNP 集合に含まれるとは言えない。このため、CNP 拡張型の場合は更新の問題が課題となっている。

4.2 静的な CNP 集合の作成

CNP 集合の作成方法は、静的に作成する場合でも動的な場合と同様である。索引構造木内の全てのオブジェクトに対して 3 章で示した方法で CNP を作成する。

4.3 検索

データを検索する際には、まず質問点からの近傍探索を実行する。次に見つかった起点オブジェクト候補それぞれに対し、起点オブジェクトからの CNP 集合の中で、起点オブジェクトから次の点までの距離が質問点から起点オブジェクトまでの距離よりも小さいものを結果のリストに加える。全ての起点オブジェクト候補に対し、同様の処理を実行して得られた結果のリストに含まれる CNP 集合が最終的な検索結果となる (図 4)。

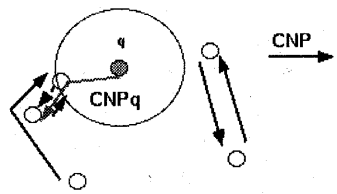

図 4: 静的索引時の検索

4.4 挿入

新しいオブジェクトを挿入する場合には、オブジェクトが格納されている索引構造木と CNP 集合の両方を更新する必要がある。オブジェクトが挿入されたときは各オブジェクトの近傍オブジェクトに変化が起こる。しかし、この変化の伝播範囲を正確に知ることは難しい。最も単純な方法は全てを再構築する方法だが、更新コストが高く実用的ではない。そのため、差分更新が必要となる。差分更新には [8] の差分更新アルゴリズムを利用する。

まず、新たに挿入される点 q からの CNP を生成する。点 q に対し、点 q が最近傍である点 reverse nearest neighbor p を求める。各 p に対し、CNP のリンク先を点 q に変える。点 q の CNP が点 p を指していない場合、点 q の CNP も CNP に加える。p を指している場合、点 q は終端オブジェクトとなり、点 q の CNP は加えない (図 5)。

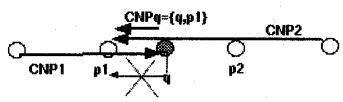

図 5: データオブジェクトの挿入

図 5 の場合、点 q から生成される CNP_q は $\{q,p_1\}$ であり、点 q を最近傍とする点 p は p_1 と p_2 の 2 点である。p_1,p_2 からはともに q へのリンクがなされるが、点 p_1 は CNP_q に含まれるため、CNP_1 は点 q が終端オブジェクトとなる。しかし、点 p_2 は CNP_q に含まれないため CNP_2 では p_1 が終端オブジェクトとなる。最後に、全ての CNP 集合の中から点 p を含む CNP を検索し、点 p から先のリンクを更新する。

4.5 削除

オブジェクトを削除する場合も索引構造木と CNP 集合の両方を更新する必要がある。削除の場合も挿入の時と同様に [8] の方法を利用する。

削除されるオブジェクトを q とすると、まず、点 q を最近傍である点 reverse nearest neighbor p を求める。各 p に対し、p からの CNP の再構成を実行する。全ての CNP 集合の中から点 p を含む CNP を検索し、点 p から先のリンクを更新する (図 6)。

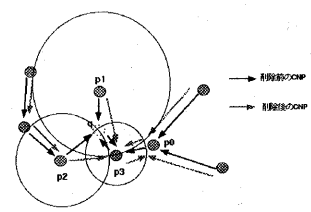

図 6: データオブジェクトの削除

図 6 の場合、削除される点 q を最近傍としている点は p_1,p_2,p_3 の 3 点である。点 q が削除されると、この 3 点からの CNP の再構成を実行する。p_1,p_2,p_3 各点からの CNP はそれぞれ、$\{p_1,p_3\}$, $\{p_2,p_3,p_0\}$, $\{p_3,p_0\}$ に更新される。最後に p_1,p_2,p_3 を自分から始まる CNP に含む点を探す。図 6 の場合は全ての点が対象となる。p_1,p_2,p_3 から先のリンク先を更新する。

5 検索結果の可視化

[1] で示された検索結果の可視化の方法を改善することで、ユーザの利用効率を増加させることを考える。

[1] では、オブジェクト間の関連度合を数値化することで、ユーザの検索目的に合った検索結果の閲覧順序のナビゲートを行った。具体的には、オブジェクトの方向性を検索結果に反映させるために、オブジェクト o_i とその o_i の最近傍点上位 s 個 $\{o_{i1},o_i i2,\ldots,o_{is}\} \in O_{is}$ との関係を、オブジェクト o_i を起点とし、その近傍点 $o_{ij},o_{ik} \in O_{is}$ のベクトル $\overrightarrow{o_i o_{ij}}$ と $\overrightarrow{o_i o_{ik}}$ とのなす角 θ の余弦 $(cos\theta)$ で表現した。これは CNP を順にたどって行くときも再帰的に行われるため、ある深さでの選択順序の決定

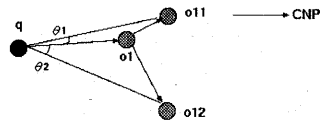

図 7: オブジェクト間関係の可視化

には便利だが、その前の段階との関係が分からないため最初に選ぶべき候補の決定に関しては不十分である。

そこで、新たに質問点からのベクトルと比較することで、選択を容易にする。質問点 q とオブジェクト o_i のベクトル $\overrightarrow{qo_i}$ を基準ベクトルとし、質問点 q と o_{ij}, o_{ik} のベクトル $\overrightarrow{qo_{ij}}, \overrightarrow{qo_{ik}}$ との余弦をとって出力する。ユーザは、$\cos\theta$ の値が大きいものを選ぶことによって、親オブジェクトである o_i の方向性に近いオブジェクトを選択することが可能になり、選択の容易性が増す。

6 結論

本稿では、近傍連鎖点列集合の構築方法に新たに静的な方法を提案した。また、静的索引時の検索と更新の方法についても示した。さらに、検索結果の可視化についての改善方法も示した。これにより、これまでの動的な構築方法に比べ検索にかかるコストは減らすことができ、また検索結果の可視化による選択の容易さの向上により、検索におけるユーザビリティの向上を図ることができたと言える。更新の問題についてはさらなる検討が必要である。

今後の課題としては、CNP 拡張型の場合の更新方法について検討が必要である。実際の検索システムとしての具体的な使用例を提示していく必要がある。さらに高次元での検索効率改善のために、現在 R*-tree を使っている索引構造木を、X-tree[6] や SR-tree[7] などに変更することも検討中である。

参考文献

[1] 田中 覚, 遠山 元道: 多次元空間における近傍連鎖点列を用いた類似検索システム, Database Engineering Workshop 2001,4B-3

[2] A. Guttman.: R-trees: A Dynamic Index Structure for Spatial Searching, Proc.ACM SIGMOD, June 1984, pp.47-57.

[3] Timos Sellis, Nick Roussopoulos, Christos Faloutsos.: THE R+-TREE: A Dynamic Index for Multi-dimensional Object, Proc. of the 13th VLDB Conf, Brighton,England, pp.507-518, Sep. 1987.

[4] Beckmann N., Kriegel H.-P., Schneider R.,Seeger B.: The R*-tree: An Efficient and Robust Access Method for Points and Rectangles, Proc.ACM SIGMOD Int.Conf. on Management of Data, Atlantic City, NJ, 1990, pp.322-331.

[5] Nick Roussopoulos, Stephen Kelly, Frederic Vincent.: Nearest Neighbor Queries, Proc, ACM SIGMOD Int.Conf. on Management of Data, San Jose, CA, pp.71-79, May 1995.

[6] Stefan Berchtold,Daniel A.Keim,Hans-Peter Kriegel.: The X-tree: An Index Structure for High-Dimensional Data, Proc. of the 22nd VLDB Conf., pp.28-39, 1996.

[7] Norio Katayama, Shin'ichi Satoh.: the SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries, Proc, ACM SIGMOD Int.Conf. on Management of Data, 13-15 May, 1997, Tucson, Arizona, pp.369-380.

[8] Flip Korn, S.Muthukrishnan.: Influence Sets Based on Reverse Nearest Neighbor Queries, Proc, ACM SIGMOD Int.Conf. on Management of Data, 2000, pp.201-212.

[9] Alexander Hinneburg, Charu C. Aggarwal, Daniel A. Keim.: What is the nearest neighbor in high dimensional spaces?, Proc. of the 26th VLDB Conference,Cario,Egypt,2000, pp.506-515.

[10] Stefan Berchtold, Daniel A. Keim, Hans-Peter Kriegel, Thomas Seidl: Indexing the Solution Space: A New Technique for Nearest Neighbor Search in High-Dimensional Space, Transaction on Knowledge and Data Engeneering, 2000, VOL12, NO.1, pp.45-57.