首藤研究室輪講第4回

4.1多次元データ共有

ファイル記述に基づくクエリを提供する方法は以，下の2通りがある．

-多次元インデックス¥re(multi-dimensional index):各属性を多次元空間の次元とし，クエリは多次元空間内を進む．CANは多次元インデックスを最初に提供したP2Pシステムであ．多次元空間のポイントクエリ¥re(point query)を提供するが，一様なハッシュを行っているため、レンジクエリは提供していない．

-複数属性インデックス¥re(multi-attribute index):属性ごとにインデックスを保持し，クエリはインデックスごとに進む．MAAN(Multi-Attribute Addressable Network)は全ての属性を一つのChordリングでインデックスすることで，複数属性インデックスを提供する．クエリは支配属性¥(dominate attribute)に基づいて進み，他の属性はフィルターとして使われる．¥(Mercury)はルーティングハブ¥(routing hub)と呼ばれる，属性ごとのChordリングを保持する．アイテムは全てのルーティングハブに送られる必要がある．

スカイラインクエリ¥(skyline query)とは，自身より全ての属性で優れているファイルが存在しないようなファイルを探索するクエリを指す．¥(SkyPeer)はスカイラインクエリを提供するために，効率的に返答を得ることが可能な部分空間スカイラインクエリを利用する．

4.1.1 VBI-Tree

VBI-Treeはバイナリツリーベースの多次元インデックスのツリー構造を採用しているフレームワークである．各ピアは間順(in-order)で隣り合う，葉ノードと内部ノードのペアを管理する．葉ノードは，ある多次元の領域に属するデータのインデックスを保持するノードである．内部ノードは，その子ノードが管理する領域全体を覆う領域のルーティングを管理するノードである．内部ノードは，以下の5種類のリンクを保持する．

-親リンク(Parent link): 親ノードへのリンク

-子リンク(Child links): 子ノードへのリンク

-隣接リンク(Adjacent links): 間順で隣り合うノードへのリンク

-近隣リンク(Neighbor links): 同じレベルで左右に2^i離れたノードへのリンク

-上方リンク(Upside links): 祖先のノードへのリンク

リンクはリンク先へのポインタだけでなく，リンク先のノードの担当する多次元領域の情報も保持する．ノードnがクエリを発行又は受信すると，クエリの範囲が自身の内部ノードの担当領域に含まれるか検査する．含まれるとき，クエリの範囲を担当する子リンクにクエリを転送する．加えて，クエリの範囲を完全に覆う担当範囲を持つもっとも近い祖先のノードaにクエリを転送する．aが存在するとき，nからaまでのパス上の各祖先a'について，nと反対側の子ノードに近隣リンクを通じてクエリを送信する．この方法により，ボトルネックを作らずに，クエリの範囲を含んでいる担当範囲をもつ全てのノードにクエリを送信することが可能である．

4.1.2 Mercury

Mercuryは複数属性インデックスを提供するために，属性ごとのインデックスを保持するハブを持つ．ハブ内のノードは円形に配列され，ハーモニック確率密度関数に基づく近隣リンクを持つ．さらに，ハブ間でのルーティングのために，各ノードは他のハブ内のノードへのリンクを持つ．

アイテムがネットワークに挿入されるとき，各属性aについてハブHaに属性値でインデックスされる．クエリは，支配属性に対応するハブを進む．各ノードは他のハブ内のノードへのリンクを保持しているため，支配属性に対応するハブに1ステップで送信することができる．

Mercuryの欠点は，データを挿入，離脱するとき，全てのハブに挿入，離脱する必要があるため，属性の数が多い場合，コストが高くなってしまうことである．

4.1.3 SSP

Skyline Space Partitioning (SSP)はP2Pネットワーク上でスカイ

ラインクエリを提供するための方法である．基本的なアイディアは，多次元空間を各ノードの担当する領域に分割することである．ノードはBATONとして配置され，担当領域の場所をz-curveの順に並べることで，1次元インデックスを提供できる．ルーティングのために，各ノードはBATONのリンクの情報に加えて，以下の担当範囲の情報を保持する必要がある．

-領域番号(Region Number): 領域のz-curve順での位置の識別子

-データ範囲(Data range): 領域が含んでいる値の範囲

-分割履歴(Split history): 領域が生成された時点からの分割を表す，値と次元の組みのリスト

-次の分割次元(Next partition dimension): 次に領域が分割されるときに，分割される次元

SSPでのスカイラインクエリは以下の4ステップで行われる．

-最初にシステムはローカルなスカイラインクエリの結果が必ず最終的なスカイラインに含まれるようなノードを探索する．このノードをSTARTERノードという．

-STARTERノードに到達すると，ローカルなスカイラインを計算し，最も大きい支配領域を持つ点Pmdを選ぶ．スカイラインの探索空間から，pmdの支配領域は除かれる．

-その後，STARTERノードは探索範囲を含んでいるノードにクエリを送信する．各ノードはローカルなスカイラインを計算し，クエリを生成したノードに結果を送信する．

-最後に，クエリを生成したノードは，受信したローカルなスカイラインの結果から全体のスカイラインを計算する．

SPPの欠点は，クエリのスピードがSTARTERノードに大きく影響されることである．STARTERノードに到達できいことや，pmdが存在しないことがあると，クエリのスピードは減少してしまう．

4.2 高次元インデックス

ファイル記述だけで完全にファイルの内容を表すことは不可能である．そのため，細かい粒度で探索可能なコンテンツベース探索が望まれている．コンテンツベース探索を提供する一般的な手法は，ファイルの特徴のベクトルを作成し，それらをインデックスする方法である．しかし，特徴のベクトルの次元は非常に大きくなり，従来の多次元インデックスでは，効率が悪い．高次元空間データをインデックスするための新しい方法は，マッピングベースアプローチ(mapping-based approach)，距離ベースアプローチ(distance-based approach)，ハッシングベースアプローチ(hashing-based approach)の3種類に大きく分類される．

-マッピングベースアプローチでは，高次元空間はインデックスの前に低次元空間にマッピングされる．複数次元クエリは，低次元でのクエリに変換されて実行される．

-距離ベースアプローチでは，リファレンスポイント(reference points)と呼ばれる，事前に定められた点からの距離に基づいて，多次元オブジェクトは直接インデックスされる．インデックスの値は，オブジェクトから最も近いリファレンスポイントへの距離と，そのリファレンスポイントの値の合計で定められる．

-ハッシングベースアプローチでは，似たオブジェクトは同じ場所にハッシュされる．クエリは，インデックスに対応するハッシュ値の周辺を探索する．この方法では常に正確な返答が得られるとは限らないが，結果のエラー率が事前に定義した値以下であることを保証する．

4.2.1 CISS

高次元インデックスを提供するために，CISS,

は多次元空間を1次元空間にマッピングするヒルベルト空間充填曲線(Hilbert Space Filling Curve , SFC)を利用し，1次元データはChordにインデックスされる．クエリは，多次元の値から1次元の値に変換されChordリング上を探索する．多次元空間の一つの領域は，1次元空間の複数の区間に変換されてしまうため，特に次元が大きいとき，クエリの効率は悪くなる．

4.2.2 ZNet

ZNetは，多次元空間を1次元空間にマッピングするためにz-curveを利用する．データ空間は四分木のような方法で再帰的に分割される．各部分空間には，Z-curveでの位置と分割のレベルに対応した一意のアドレスが割り当てられる．各ノードは各部分空間とその空間内に位置するデータを担当する．ノードは担当する領域のアドレスの順にSkip Graphに並べられる．ノードがクエリを発行又は受信すると，探索範囲を含んでいる部分空間のアドレスを調べ，その部分空間を担当しているノードにクエリを転送する．空間の分割の情報が不足しており，アドレスが完全には知ることが不可能であるときも，接頭辞を知ることは可能である．この場合，探索範囲に近い隣接ノードにクエリを転送する．

4.2.3 M-Chord

M-Chordは，高次元空間での類似検索 (similarity search)を提供する．M-Chordは以下の2段階で多次元インデックスを実現する．

1.高次元データを1次元データにマッピングするために，iDistanceを利用する．iDistanceは，全体に知られた基準点 (reference points)を利用しデータ空間を分割し，データを最も近い基準点からの距離によってインデックスする．

2.1段階目で得られた1次元の値をChordリング上にインデックスする．

iDistanceでは，データ空間を事前に分割の集合S = {P1,P2,...Pn}に分ける必要がある．各分割Piは基準点Oiと半径riによって表される．各pi(oi,ri)が互いに重ならない区間[i·c, i·c + ri]に相当するように定数cを選ぶ．(keyOi =　i　·　c はOiが対応する1

次元の値)データオブジェクトDがシステムに挿入されると，最初にデータオブジェクトから最も近い基準点Ojを調べ，Dのインデックスの値をj\*c + dist(D,Oj)と計算する．そして，1次元インデックスの値でChordリングに挿入する．

oを中心，rを半径とするレンジクエリQR(o,r)を行うとき，最初に検索範囲を含む分割を調べる．その後，各分割Pi(Oi,ri)について，1次元レンジクエリq[keyOi + dist(Oi,o) − r,max(keyOi + dist(Oi,o) + r,keyOi + ri)]をChordリング上で行う．

oを中心，k探索するオブジェクトの個数とするk近傍(kNN)クエリは以下の2段階で行う．

1.低コストな発見的手法でOから近いk個のオブジェクトを見つけ，それらのオブジェクトとoとの距離の最大値sigumaを求める．

2.レンジクエリQR(o,siguma)を行い，その結果の中から，oから近いk個を選ぶ．

オブジェクトを分離してインデックスする必要があるため，オブジェクトの個数が多いとき，インデックスのコストが高くなってしまう．

4.2.4 SIMPEER

インデックスのコストが高くなることを避けるため，SIMPEERはインデックスの前にオブジェクトを要約し，その要約のみをインデックスする．以下の3つのレベルのインデックス構造を利用する．

-最も低いレベルでは，各ピアは自身の共有オブジェクトをインデックスする．さらに，自身の共有オブジェクトをクラスタとして，その要約を作成し，自身を担当しているスーパピアに送信する．

-スーパピアのレベルでは，担当しているピアから送られてきたクラスタの要約をインデックスする．また，それらのクラスタの要約でハイパークラスタを作り，他のスーパピアに送信する．

-最も高いレベルでは，スーパピアは自身のハイパークラスタと他のスーパピアから受信したハイパーピアのインデックスを作成する．

ピアがクエリを発行するとき，スーパピアにクエリを送信する．スーパピアは，ローカルなクラスタの要約やハイパークラスタのインデックスに基づき，担当ピアや他のスーパピアにクエリを転送する．

クラスタの要約をインデックスするために，SIMPEERはiDistanceと同じように，全体に知られた基準点を利用しデータ空間を分割する．最初にクラスタの中心に最も近い基準点oiを持つ分割Piを調べ，クラスタの最も遠い点を基準点oiに基づいて，1次元のインデックスの値に変換する．レンジクエリは，探索範囲と重なっている部分空間を含む各分割piに対し，クエリの範囲のうち基準点oiに最も近い点から，分割piの境界までを探索することで実行する．

SIMPEERには2つの欠点が存在する．1つは，基準点とクエリ範囲の距離が近いとき，クエリ範囲の大きさに関わらず，広い範囲を探索する必要がある点である．もう1つは，分割はクラスタ全体を覆う必要があるため，探索空間が大きくなり，探索コストが高くなる．

4.2.5 LSH Forest

LSHホレスト (LSH Forest)は，複数のLSH木(LSH Tree)を含むインデックス構造である．このシステムは，インデックスのために局所性鋭敏型の一群を使用する．オブジェクトOはl個のLSH木にインデックスされる．各LSH木でOにはx桁の識別子が割り当てられる．識別子は，Hから選んだh1(),h2(),hx()を利用して，h1(O),h2(O),...,hx(O)と並べたものとなる．識別子の各桁がLSH木の根からのパスを表し，Oを示す葉の位置が決まる．

類似クエリを実行するには，クエリオブジェクトの識別子を同様に生成し，l個のLSH木で接頭辞が最長一致する葉をトップダウンに探す．それらの葉からボトムアップに類似のオブジェクトを集める．その中から最もクエリオブジェクトに近いオブジェクトを返す．