QUILTS: クエリアウェアでスキュートレラントな空間充填曲線を用いた多次元データ分割フレームワーク

西村　祥治[†,††]　　横田　治夫[†]

†† 日本電気株式会社　〒 216–8666 神奈川県川崎市中原区下沼部 1753
† 東京工業大学情報理工学院　〒 152–8552 東京都目黒区大岡山 2-12-1
E-mail: †s-nishimura@bk.jp.nec.com, ††yokota@cs.titech.ac.jp

あらまし　近年、データ分析の対象となるデータ量の増大のため、データアクセスのボトルネックを解消するデータ管理技術が重要になってきている。このため、データ分析に必要なデータだけをアクセスするデータスキッピング技術が注目を集めている。基本的なアイデアは、データをページ分割し、クエリによって取得されるデータを含むページだけアクセスすることである。アクセスするページを最小にする必要があるが、与えられたクエリパターンやデータ分布を用いて最適解を求めることはＮＰ困難であることが知られている。そこで、我々は空間充填曲線を用いたデータ分割手法を提案する。我々は空間充填曲線とクエリパターンとデータ分布の関係をモデル化した。そして、そのモデルに基づき、対象となるクエリパターンに対して、データ分布に偏りに対して頑健な空間充填曲線を設計する手法を提案する。我々はこの手法を基に多次元データ分割フレームワークを構築し、ＤＷＨやＧＩＳを対象とした実世界のデータを用いた評価を行った。その結果、ページアクセス数を従来の空間充填曲線を用いるより 1 ケタ少なくできることを確認した。

キーワード　多次元インデックス, 空間充填曲線, データ分散

1. はじめに

近年、ウェブサイトのログや様々な機器からのセンサーデータなど大量データの収集と分析が広まっている。多くのデータ分析では、その収集されたデータの一部だけを用いる。そこで、一部のデータだけを効率的に取り出すためデータスキッピング [14], [18] 手法が注目を集めている。基本的な考え方は、データをページに分割し、クエリによって取得されるデータを含むページだけをアクセスする。このクエリ性能はページアクセス数に比例するため、与えられたクエリパターンやデータ分布に対してページアクセス数を削減するようなデータ分割をすることが重要になる。しかしながら、属性を複数持つ多次元データに対して、最適なデータ分割を求めることは NP 困難であることが知られている [18]。

Sun らは最適に近いデータ分割を得るため、クエリパターンやデータ分布を考慮したクラスタリングアルゴリズムを用いる手法を提案している [18]。この手法によりページアクセス数を劇的に削減することに成功している。しかしながら、この手法はクエリを実行する前にすべてのデータに対してクラスタリングアルゴリズムを適用する必要がある。

別のデータ分割方法は空間充填曲線を用いる方法である [8], [13]。この方法は多次元データを空間充填曲線上の点に一対一で対応付けることで、一次元データ化し、それを範囲分割するものである。一次元化したデータを B-Tree や HBase など範囲分割するキーバリューストアのキーとして格納することで、データの挿入や更新時に適切な範囲になるようにデータを分割することができる。しかしながら、この方法では、どの空間充填曲線を用いて多次元データを一次元データ化するかによって、クエリ実行時にアクセスするページ数が変化する。これまで、どの空間充填曲線を用いるべきかについて幅広く研究がされてきた [8], [17]。しかし、多くの研究は一般化された場合であるアドホックなクエリパターンを対象としており、特定のクエリパターンに対してはあまり注意が払われてこなかった。また、空間充填曲線を選択する際、データ分布を考慮することも重要になる。しかしながら、我々の知る限り、データ分布と空間充填曲線の関係についてほとんど議論されていない。

我々は空間充填曲線とクエリパタンおよびデータ分布の関係について分析をした。その結果、与えられたクエリパターンに対して、データ分布の偏りに対して頑健な曲線を選ぶことで、クエリ実行時にアクセスするページ数を抑えられることができることを見出した。このような曲線をクエリ特化でスキュー耐性がある曲線 *(query-aware and skew-tolerant curve)* と呼ぶ。

そこで我々はこの曲線を基にした多次元データ分割のフレームワークである QUILTS(**QU***ery-***I***ntensive* **L***inearization* **T***olerating data* **S***kew*) を提案する。その概要を図 1 に示す。このフレームワークは、凝集性に基づくコストモデル、ビット混合曲線族、曲線設計法の 3 つの要素から構成され、入力としてクエリパターンを与えるとそのクエリパターンに対してスキュー耐性がある曲線を出力として得ることができる。我々はこのフレームワークに基づいて試作した。そして、実世界のデータを用いたアプリケーションで評価した結果、このフレームワークが導出した曲線は、従来よく知られている曲線に対して、1 桁以上ページアクセス数を削減できることを確認した。

本論文の構成は以下のとおりである。2 章では空間充填曲線

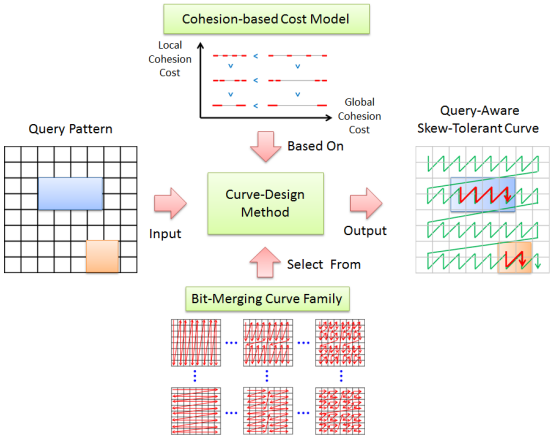


図 1　QUILTS の概要

に関する関連研究について紹介する。３章では前提となる概念について説明する。４章ではクエリ特化でスキュー耐性がある曲線を用いたデータ分割手法を提案する。５章では評価を行い、有用性を示す。６章では結論を述べる。なお、本研究についての詳細は文献 [16] を参照されたい。

2. 関 連 研 究

2.1 空間充填曲線に基づく索引

多次元索引を実現する方法として空間充填曲線 [1] を用いる手法 [15] が知られている。リレーショナルデータベース (RDB) で使われている B-Tree と組み合わせた手法 [6], [8], [20] や、HBase のような範囲分割のキーバリューストア上で多次元索引を実現する手法 [4], [13], [22] など広く適用されている。例えば、RDB で広く使われている複合インデックスは図 2(a) のような軌跡を描く C-カーブを用いた索引である。属性の値を連結することでデータの並び順を決定することができるため、属性ごとの定義域の違いがある場合でも適用することができる。しかし、高い性能を達成するには、クエリでの属性ごとの選択性を意識して属性の優先度を決定することが重要である [19]。

Z-カーブ (図 2(b)) [12] やヒルベルトカーブ (図 2(c)) [3] は特に空間索引で広く使われている空間充填曲線である。正方形の構造が階層的に埋め込まれていることから、位置情報のように各属性が同じ距離基準を持つような応用に適合しやすい。一方で、この特徴から定義域が異なる属性からなる多次元データに適用する際、ゼロで埋めるなどしてデータ表現上の定義域を揃える必要がある。この結果、不必要にデータ量が増えることになる。そこで、これらのデータ増加を抑える曲線が幾つか提案 [2], [7] されているが、クエリ実行時のアクセスコストに主眼をおいていない。

2.2 空間充填曲線の性質の分析

上記以外にも様々な空間充填曲線が提案される一方で、最適な空間充填曲線が持つべき性質およびそれを計測する指標に関する研究も広く行われている。例えば、Mokbel らは元の空間上で近傍にある二点を、空間充填曲線上でも近傍に写像する局所保存性を評価する指標として irregularity を提案している [9], [10]。また、Moon らはディスクシーク時のアクセス

(a) C-カーブ　(b) Z-カーブ　(c) ヒルベルトカーブ

図 2　代表的な空間充填曲線

連続性を評価する指標として clustering number を提案している [11]。そして、Pan らはヒルベルトカーブが一般的な場合における clustering number を最善にするカーブの一つであると証明している [21]。

これらの研究は一般的な場合に注目している。しかしながら、実際の応用では、暗黙的な上限値や下限値や、典型的な集約単位が存在する。このような制約条件下で、適切な空間充填曲線を選択する上で注目すべき性質やその指標についてはあまり考慮されてこなかった。また、データ分布はクエリ性能に影響を与えるにも関わらず、データ分布に対する空間充填曲線の性質は我々が知る限り、あまり考慮されていない。

3. 前 提 知 識

3.1 空間充填曲線による多次元インデックス

空間充填曲線とは多次元空間上の点を漏れや抜けなく一筆書きでたどる曲線である。この曲線がたどる順番により多次元空間上の点が順序付けられる。すなわち、多次元空間上の点が一次元の点へとマッピングすることができる。ここではこのマッピングされた点のことをキーと呼ぶ。このキーの値によって整列して格納されたデータ構造を空間充填曲線による多次元インデックスと呼ぶ。例えば、このキーを B-Tree や HBase などの範囲分割型のキーバリューストア (KVS) に格納することで容易に実現できる。このとき、データはキーの範囲に応じてページに分割して格納される。B-Tree や HBase など多くのデータ構造では、キーの範囲は各ページがだいたい同じ量になるように調整される。

このような多次元インデックスに対して多次元範囲クエリの実行を模式的に示したものが図 3 である。多次元範囲クエリで指定された領域は、空間充填曲線上で幾つかの連続する区間の集合に写像される。多次元範囲クエリの実行では、まずこれら区間と重なるページ集合を抽出する。そしてこのページ集合をアクセスし、クエリ範囲内にあるデータを結果として報告する。この例では、ページ 3,5,7,9 をアクセスするため、ページアクセス数は 4 となる。

ここで重要なことは、多次元範囲クエリの性能は、ページアクセス数に比例するという点である。つまり、クエリ実行時にアクセスしなければならないページ数を抑えることができるような空間充填曲線を選ぶことが我々の目的である。

3.2 空間充填曲線とクエリパターン

まず、空間充填曲線とクエリパターンの関係について整理する。クエリパターンとは、多次元範囲クエリの形状の観点でのクエリの種類分けで、各属性を選択する範囲の幅の直積として

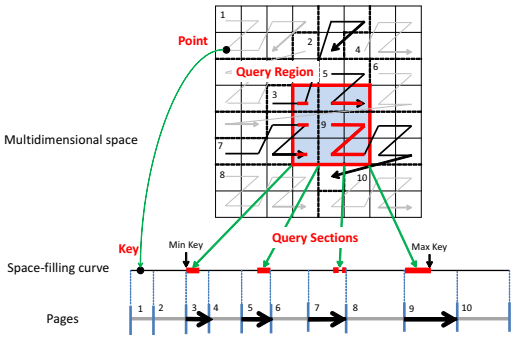
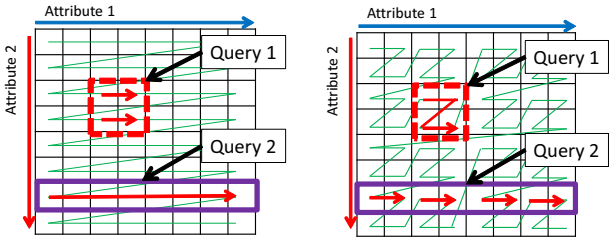


図 3　Mapping Between Multidimensional Space and Space-Filling Curve



(a) C-curve　(b) Z-curve

図 4　Space-Filling Curves and Query Patterns

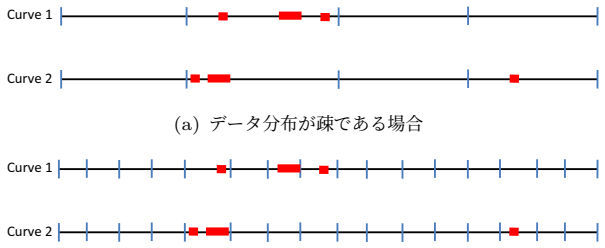
表現する。例えば、次元数が d あり、属性 i の選択幅が W_i とすると、そのクエリパターンは $W_1 \times W_2 \times ... \times W_d$ と表現する。

クエリパターンがページアクセス数に与える影響について図**??**を用いて模式的に分析する。ここでは空間充填曲線として C-カーブと Z-カーブを対象とし、クエリパターンとして各属性の選択幅が等しい Query 1（点線で囲まれた矩形）と各属性の選択幅が極端に異なる Query 2（太線で囲まれた矩形）を考える。それぞれのクエリパターンを表す矩形の中での曲線の軌跡の連続性に注目して分析する。すると、C-カーブは、Query 1 では非連続が、Query 2 では連続である。一方、Z-カーブは、Query 1 では連続であるが、Query 2 では非連続である。一般的に、クエリ範囲内での曲線の軌跡が連続するほど、クエリで取得するデータが同一のページに含まれやすい。逆に非連続になるほど別のページに分かれやすい。このことから、空間充填曲線ごとに適合しやすいクエリパターンがあるといえる。そこで、あるクエリパターンと相性がよい空間充填曲線を**クエリ特化した曲線 (query-aware curve)** と呼ぶ。

3.3　空間充填曲線とデータ分布

次に、空間充填曲線とデータ分布の関係について整理する。

ここではページに容量があり、データの挿入や削除により、ページ内のデータの件数が一定量を超えて変動した場合、一定に保つようにページの分割あるいは結合により調整が行われることを仮定する。これは、RDB 中の B-Tree や範囲分割型の KVS である HBase などで一般的にサポートされている機能である。この時、データ分布が密である領域にあるページはページの分割が進み、ページがカバーする範囲が狭くなる。一方でデータ分布が疎である場合は、逆にページがカバーする範囲が広くなる。



(a) データ分布が疎である場合

(b) データ分布が密である場合

図 5　Query Section Layout and Data Distributions

今、2 つの空間充填曲線 Curve 1, 2 があり、ある多次元範囲クエリに該当する空間充填曲線の区間がそれぞれ図 5 の赤太線であったとする。データ分布が疎な場合、Curve 1 は一つのページにすべての区間が含まれているため、ページアクセス数は 1 である。一方 Curve 2 のページアクセス数は 2 である。データ分布が密な場合、Curve 1 でのページアクセス数は 3 であるが、Curve 2 では 2 である。このことから、データ分布の違いによって、ページアクセス数の観点で最適となる空間充填曲線が変わり得ることが推察できる。

実際のデータセットはデータ分布が疎である領域もあれば、密である領域も存在する。また、あるクエリパターンによる領域がある空間充填曲線上に写像される区間の配置パターンは、領域の起点となるビットのアライメントが同じであれば、同じパターンになる。このことから、あるクエリパターンに対して、データ分布の疎密に関わらず、ページアクセス数を低く抑えられるような空間充填曲線を利用することが望ましい。そこで、このような曲線を**スキュー耐性がある曲線 (skew-tolerant curve)** と呼ぶ。

4.　提 案 手 法

以上の議論から、与えられたクエリパターンに対して、データ分布の疎密に関わらずページアクセス数を低く抑えられる空間充填曲線を用いて多次元インデックスを構成すると高い性能を達成すると考えることができる。以下では、このような空間充填曲線の性質を捉えるコストモデルおよびこの指標の観点で最適な空間充填曲線を設計する手法について説明する。

4.1　凝集性に基づくコストモデル

空間充填曲線のデータ分布に対する性質を捉えるコストモデルを設計するために、もう一度図 5 を注意深く観察する。データ分布が疎である場合において、Curve 1 が Curve 2 よりもページアクセス数が少なくなった理由は、クエリ領域の空間充填曲線上への写像となる区間全体がコンパクトにかたまっているからである。このため、ページがカバーする範囲が比較的広い場合、一つのページで区間全体がカバーされやすくなる。一方で、データ分布が密である場合において、Curve 2 が Curve 1 よりもページアクセス数が少なくなった理由は、空間充填曲線上に写像されたクエリ領域の区間が局所的にコンパクトにかたまっているからである。このため、ページがカバーする範囲が比較的狭い場合、少ないページ数で区間全体がカバーされや

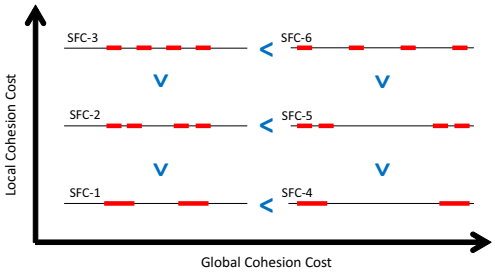


図 6　凝集性に基づくコストモデル

すくなる。ここで注目すべき点は、ある区間とある区間が空間充填曲線上で遠く離れ、これらの区間の間に多数のページが存在することになったとしても、クエリ実行におけるページアクセス数には影響を与えない点である。

以上から、図 6 に示すように、大域的な区間の凝集性と局所的な凝集性の二つの観点から空間充填曲線を評価するコストモデルを提案する。

［定義 1］（大域的凝集性） 大域的凝集性 C_G とは、空間充填曲線上に写像されたクエリの区間全体のまとまり具合を評価する指標である。クエリの区間と区間の間となる間隔の長さの総和として定義する。すなわち、i 番目の間隔の長さを n_i とすると、

$$C_G = \sum_i n_i \tag{1}$$

である。

［定義 2］（局所的凝集性） 局所的凝集性 C_L とは、空間充填曲線上に写像されたクエリの区間の局所的なまとまり具合を評価する指標である。クエリの区間と区間の間となる間隔の総和に対する各間隔の割合のエントロピーとして定義する。すなわち、i 番目の間隔の長さを n_i とすると、

$$C_L = -\sum_i \frac{n_i}{\sum_i n_i} \log \frac{n_i}{\sum_i n_i} \tag{2}$$

である。

空間充填曲線上に写像された区間ではなく、区間と区間の間隔に着目する点がポイントである。なぜなら、クエリ範囲内にあるデータ件数が多くてページアクセス数が多くなることは問題ではなく、クエリ範囲外にあるデータによって不必要にページアクセス数が多くなることが問題であると考えるからである。また、局所的凝集性でエントロピーを用いるのは、間隔の数の少なさと間隔の長さの偏りを評価したいからである。間隔の長さの総和が一定で間隔の数が同じであるならば、各間隔の長さがすべて等しいよりも著しく偏りがある方が、区間同士が局所的にかたまっているといえるからである。

あるクエリパターンに対して、ある空間充填曲線の大域的凝集性および局所的凝集性が共に小さい値である時、データ分布の疎密に関わらずページアクセス数を抑えることが期待できる。このような曲線を**クエリに特化し、スキュー耐性がある曲線 (query-aware and skew-tolerant curve)** と呼ぶ。また、これらを総合的に評価する指標として、総合凝集性を定義する。

［定義 3］（総合凝集性） 総合凝集性 C_T とは大域的凝集性と局所的凝集性をかけ合わせたものとして定義する。すなわち、i 番目の間隔の長さを n_i とすると、

$$\begin{aligned} C_T &= C_G \cdot C_L \\ &= (\sum_i n_i)(-\sum_i \frac{n_i}{\sum_i n_i} \log \frac{n_i}{\sum_i n_i}) \\ &= (\sum_i n_i) \log(\sum_i n_i) - \sum_i (n_i \log n_i) \end{aligned} \tag{3}$$

である。

あるクエリパターンに対して適切な空間充填曲線を選択する際、この総合凝集性をコストとして曲線を評価する。

4.2　C-カーブと Z-カーブの凝集性コスト分析

様々なクエリパターンに対して Z-カーブと C-カーブの凝集性コストを計測し、コストモデルの妥当性を評価するとともに傾向について評価する。ここでは、7 つのクエリパターン $(2 \times 128, 4 \times 64, 8 \times 32, 16 \times 16, 32 \times 8, 64 \times 4, 128 \times 2)$ についてそれぞれ凝集性コストを計測した。その結果を図 7 にまとめる。

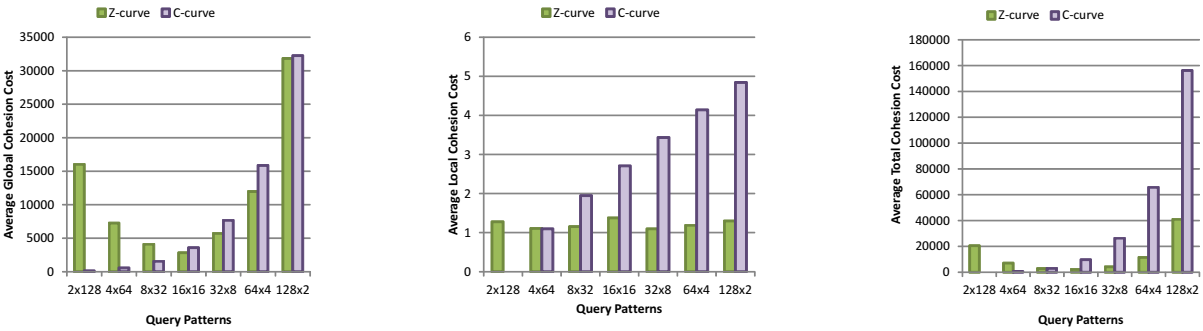
まず、C-カーブの結果について分析する。すると大域的凝集性、局所的凝集性とも 2×128 のときが最小になり、128×2 となる。総合凝集性はこれらを掛け合せたものであるため、同様の傾向である。これはちょうど、複合インデックスにおいて範囲の選択性が高い属性を優先して結合した方が性能が高く、それを逆の結合順にすると著しく性能が悪化するという結果と一致している。

次に Z-カーブの結果について分析する。Z-カーブの場合、大域的凝集性は 16×16 の正方形型のクエリパターンのときに最小になる。一方、局所的凝集性はどのクエリパターンでもあまり変化しない。これらを掛け合せた総合凝集性では 16×16 のクエリパターンのときに最小になる。以上の結果から、このコストモデルは妥当であると推測できる。

最後に興味深い結果について報告しておく。8×32 のクエリパターンに着目する。大域的凝集性に関しては C-カーブが、局所的凝集性に関しては Z-カーブがよい性質を示している。そして、総合凝集性は両カーブとも同程度の数値である。ページの都合で実験結果の詳細を省略するが、データ件数が少なく、ページがカバーする範囲が広い場合、すなわち、データ分布が疎である場合は C-カーブのほうが、逆にデータ件数を増やし、ページがカバーする範囲が狭い場合、すなわち、データ分布が密である場合は Z-カーブのほうが少なかった点を報告しておく。

4.3　曲線設計法

ここでは、任意のクエリパターンが与えられた時、適切な空間充填曲線を設計する手法について提案する。前節では、属性の範囲幅が大きく異る、全て等しいという極端なクエリパターンに対して、それぞれ C-カーブおよび Z-カーブが適合することを見てきた。では、これらの中間的なクエリパターンに対しては、どのような空間充填曲線を選択すればよいだろうか。以下では、C-カーブと Z-カーブを一般化し、C-カーブや Z-カー



(a) Global Cohesion Cost (b) Local Cohesion Cost (c) Total Cohesion Cost

図 7 Cohesion Cost Analysis of Query Patterns with Constant Area Size

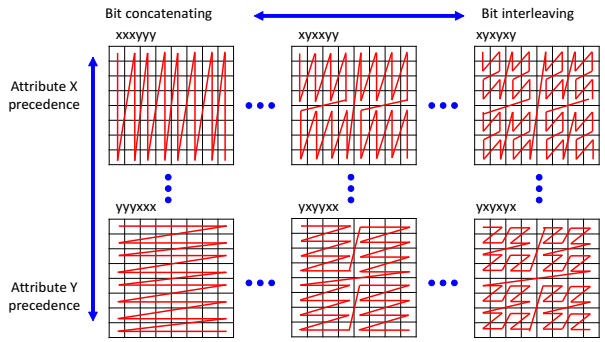


図 8 ビット混合曲線族の例

ブの中間的な性質を持った空間充填曲線の構成方法について説明する。そして、与えられたクエリパターンに対して、適切な空間充填曲線を選択するヒューリスティックを提案する。

4.3.1 ビット混合曲線族

C-カーブと Z-カーブは一見すると全く別物の空間充填曲線に見える。しかし、その構成法に着目するととても良く似ている。C-カーブは、属性値をビット列としてみた時、これらのビット列を連結することで構成する。一方で、Z-カーブは、ビット列を互い違いに混ぜ合わせることで構成する。例えば、4 ビットの二次元値 $\langle \mathtt{x_1x_2x_3x_4}, \mathtt{y_1y_2y_3y_4} \rangle$ を考える。C-カーブにおいてこの二次元値の空間充填曲線上での順序は、$\mathtt{x_1x_2x_3x_4y_1y_2y_3y_4}$ で定義できる。一方で、Z-カーブの場合は、$\mathtt{x_1y_1x_2y_2x_3y_3x_4y_4}$ で定義できる。つまり、これらの空間充填曲線の構成上の違いは、属性値であるビット列を混合する順序の違いであるといえる。そこで、このビット混合順序の観点で一般化した空間充填曲線の集合を**ビット混合曲線族 (bit-merging curve family)** と呼ぶ。図 8 にその例を示す。

ビット混合曲線族はビット混合順により一意に曲線を識別することができる。一方でビット混合順で曲線を表すにはスペースを使いすぎてしまうため、ランレングス符号化した表記を用いる。例えば、$\mathtt{x_1y_1y_2y_3y_4x_2x_3x_4}$ は、x1y4x3 と表記する。また、カッコを付けてビット列をグループ化してもよい。例えば、Z-カーブ $\mathtt{x_1y_1x_2y_2x_3y_3x_4y_4}$ は、(xy)4 と表記する。

4.3.2 クエリパターンとビット混合順

ビット混合曲線族を導入することにより、空間充填曲線の設計空間は爆発的に増加する。設計空間の広さは、属性 i のビット数が n_i とすると、n_i に関する多項分布係数になるため、

$$\frac{(\sum_{0 \leqq i < d} n_i)!}{\prod_{0 \leqq i < d}(n_i!)}$$

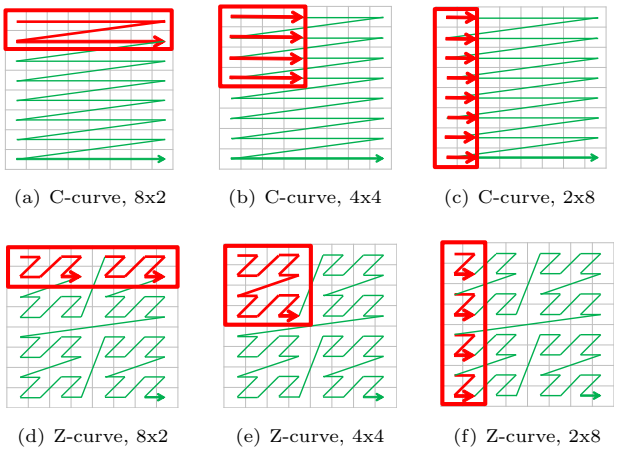
になる。このため、あるクエリパターンに対してありとあらゆるビット混合曲線族の凝集性コストを求め、最適な空間充填曲線を選定することは非現実的である。そこで、クエリパターンとビット混合順について定性的に分析し、ヒューリスティックを設計することを試みる。

まず、4.2 節での分析をビット混合順の観点で再度検討してみる。図 9 にクエリパターンと、C-カーブと Z-カーブの軌跡の関係を整理する。C-カーブは 8×2 の、Z-カーブは 4×4 のクエリパターンの時、クエリ範囲内の軌跡が連続している。これらのクエリパターンを $2^3 \times 2^1$、$2^2 \times 2^2$ と表記を変えて、それぞれのビット混合順を比較する。すると $2^3 \times 2^1$ のべき数の 3 と 1 に着目すると C-カーブのビット混合順の y3x3 において、x の 3 ビットと y の 1 が下位に配置されている。同様に $2^2 \times 2^2$ のべき数の 2 と 2 に着目すると Z-カーブのビット混合順の yxyxyx において、x の 2 ビットと y の 2 が下位にかたまっている。つまり、クエリパターンの範囲幅を 2 のべき数で表記したときのべき数に相当するビットがビット混合順において下位に配置されているとクエリ範囲内の軌跡が連続する。ここから、クエリパターンの範囲幅を 2 のべき数で表記したときのべき数に相当するビットがビット混合順において下位に配置されているビット混合曲線は、このクエリパターンに対して総合凝集性コストを低く抑えるのではないかと推察できる。

そこで、80 種類以上のビット混合曲線を機械的に生成し、クエリパターン $8 \times 32 = 2^3 \times 2^5$ に対する総合凝集性を計算した。総合凝集性のコストが低いもの順に並べた結果上位 25 の曲線を図 10 にまとめる。この時、x に関しては 3 ビット分、y に関しては 5 ビット分がビット混合順において下位に配置されているものは濃い色で示す。すると上記で推察した曲線が数多く上位に占めていることが分かる。

4.3.3 曲線設計法

以上の分析結果を踏まえて、ヒューリスティックによる曲線設計法を提案する。



(a) C-curve, 8x2 (b) C-curve, 4x4 (c) C-curve, 2x8

(d) Z-curve, 8x2 (e) Z-curve, 4x4 (f) Z-curve, 2x8

図 9 Query Patterns and Trajectory Continuity

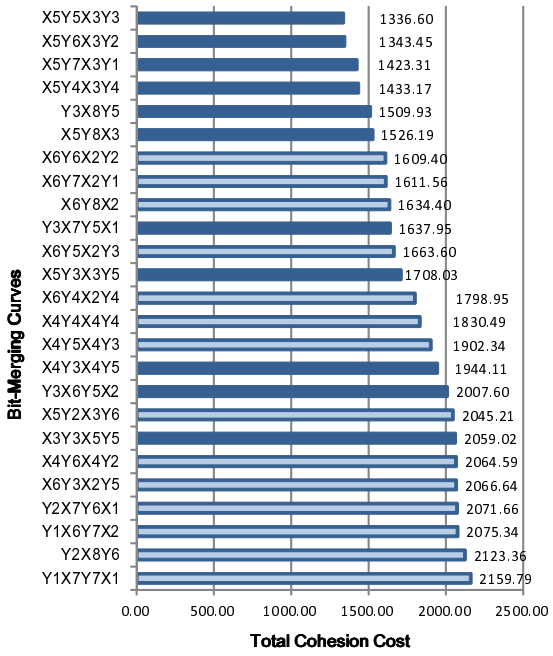


図 10 Top 25 Candidate Curves for 8 × 32 Queries

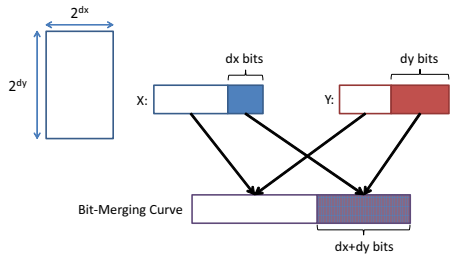


図 11 Curve for Single Query Pattern

a) クエリパターンが単一である場合

まず、クエリパターンの各属性における選択幅のビット数を求める。そして、そのビット数に相当するビットをビット混合順序において下位に配置する (図 11)。

b) クエリパターンが複数ある場合

まず、各クエリパターンの各属性における選択幅のビット数を求める。属性ごと選択幅のビット数の最大値と最小値を求める。次に、これらの値で各属性を 3 つの部分に分解し、属性間で部分ごとにビットを混合する (図 12)。

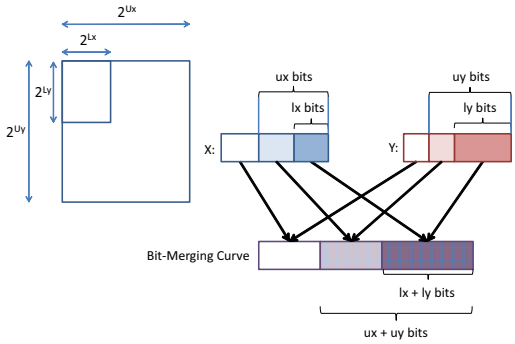


図 12 Curve for Multiple Query Patterns

表 1 Data Model for Retail Sales Analysis

Dimension	Layered Attribute	Bit Length	Distribution
Date		32	Poisson
Product		(32)	
	Department	8	Zipf
	Category	10	Zipf
	Brand	14	Zipf
Store		(32)	
	Region	6	Normal
	Store	10	Normal

表 2 Aggregation Unit and Query-Range Width

Date	Product	Store
day (16.4 bits)	category (14 bits)	region (10 bits)
week (19.2 bits)	department (24 bits)	
month (21.3 bits)		

5. 評　　価

提案するカーブの有用性を、データウェアハウスおよび地理情報システムを題材とした評価実験を行う。

5.1 データウェアハウスを題材とした評価

ここでは、文献 [5] で紹介されているケーススタディを基にデータモデルを定義した。表 1 にその詳細をまとめる。このデータセットには Date, Product, Store の 3 つの次元がある。各次元は表に示すデータ分布に従う。この時、各次元ごとに偏りの度合いが変わる点に注意する。この応用での典型的なワークロードは集約演算である。各次元はそれぞれ階層的な集約単位を持ち、その任意の組み合わせをこの応用における典型的なクエリパターンとした。各次元における集約単位およびその選択幅を表 2 にまとめる。

これらの情報をもとに空間充填曲線を設計する。Date, Product, Store 属性に対応するビットを d, p, s とする。提案する曲線は p8s6d12p10d20s10 である。比較対象の曲線として RDB 実装で典型的な C カーブとして s16d32p32 を、Z カーブとして (sdp)16(dp)16 と (dp)16(sdp)16 の 2 種類を用意した。さらに実用上よく使われるインデックスの実装として R*-Tree も用

表 3　Aggregation Units and Query-Range Widths

Longitude & Latitude	Time
0.0001 (6.6 bits)	hour (11.8 bits)
0.001 (10.0 bits)	day (16.4 bits)
0.01 (13.3 bits)	week (19.2 bits)
0.1 (16.6 bits)	month (21.3 bits)

意した。

実験のため、上記のデータモデルに従って、1 億件のデータを生成した。ページの許容量を 1000 件に設定してデータをロードしたところ、各実装ともおよそ 20 万ページに分割された。各次元の任意の集約単位の組み合わせごとに 1000 種類のクエリを生成し、クエリを実行し、平均ページアクセス数で評価した。その結果を図 13 にまとめる。縦軸は平均ページアクセス数であり、対数目盛となっている。横軸はクエリパターンである。

提案カーブはすべてのクエリパターンにおいて最も良い結果を得ることができた。2 番めに良かった Z カーブ 1 と比較しても最大で 6 分の 1 にページアクセス数を抑えることができた。提案カーブは想定するクエリパターン全てに対してなるべく適合するように設計したため、全クエリパターンに渡ってページアクセス数を抑えることができた。一方で、Z カーブ 1 は各次元の選択幅が近いクエリパターンでは提案カーブに迫るものの、選択幅が大きく異なるクエリパターンではページアクセス数が増加した。

意外だったのは R*-Tree の結果である。選択幅が小さいときは相対的によい結果である。しかし、選択幅が大きくなるに連れてページアクセス数が増加した。この理由としては、Product 次元が他の次元と比べてデータ分布の偏りが大きいことが影響しているためと考えられる。すなわち、Product 次元のデータ分布の偏りが大きいため、この次元の方向にページ分割がされやすくなったと考えられる。この結果、Product 次元の選択幅が大きくなるにつれ、顕著にページアクセス数が増加したと考えられる。提案カーブはデータ分布の偏りに対して寛容な性質を持っているため、R*-Tree のような結果に陥らなかったと考える。

5.2　地理情報システムを題材とした評価

この評価実験では、NYC Yellow Taxi の 2015 年一年間の位置情報ログのデータセットを用いた評価をした。このデータセットは 1.5 億件のレコードが含まれていた。しかし、このうち 2% は GPS 機器の故障と思われる無効なデータが含まれていた。このデータセットは、タクシーが乗客を乗せた／降ろした位置（緯度と経度）と時刻、乗客数、支払額などの情報が含まれている。

この評価実験では、時空間統計分析をワークロードとして想定した。すなわち、場所や期間あたりの乗客数、売上高を計算するものである。範囲検索の対象となる次元は、緯度、経度、時刻の 3 次元である。各次元の集約単位は実用上意味のある単位を選び、表 3 のとおりである。これらの選択幅の任意の組み合わせを評価対象のクエリパターンとした。また、この集約単位に関して、位置（緯度、経度）と時刻とで選択幅および選択幅の増分が異なる点に注意する。すなわち、位置に関する次元のほうが時刻より選択幅が相対的に小さく、位置に関する次元の選択幅の増分は均一であるが、時刻はそうではない。

提案する曲線として、次元間の選択幅の相違だけを考慮した (xy)5(txy)27t5(提案カーブ 1) と、選択幅の増分の相違も考慮して設計した曲線 (xy)5(xyt)10t2(xy)3t3(xy)4(xy)3t5(xyt)7t5(提案カーブ 2) を用意した。比較対象として、Z-カーブを時刻優先の (txy)32(Z カーブ 1) と位置優先の (xyt)32(Z カーブ 2) の 2 種類を用意した。また、他の種類の空間インデックスとして R*-Tree を用意した。

データをそれぞれの実装に対してロードしたところ、それぞれおよぼ 20 万ページに分割された。それぞれのクエリパターンに対して 1000 種類のクエリを生成し、クエリを実行した際の平均ページアクセス数で評価した。その評価結果を図 14 にまとめる。

クエリパターンをより考慮した提案カーブ 2 が 10 種類のクエリパターンにおいて最も良い結果を得た。一方でこの曲線は、時間次元の選択幅が小さく、空間次元の選択幅が大きいクエリタイプでは結果が悪かった。この原因としては、すべてのクエリパターンにおいて、クエリパターンが関係するビットを下位に配置するようなビット混合順序を定義することができないことに起因している。このことから、複数のクエリパターンでビット混合順序が干渉する場合、どのクエリパターンの性能向上させるかを優先付ける必要がある。

6. おわりに

我々は、クエリ特化でスキュー耐性がある空間充填曲線を用いた多次元データ分割フレームワークを提案した。このフレームワークでの技術的な貢献は、(1) 空間充填曲線のクエリパターンおよびデータ分布に対する性質を明らかにするモデル、(2)Z-カーブと C-カーブを一般化し、空間充填曲線の設計空間を爆発的に拡張したビット混合曲線族、(3) ビット混合曲線族を用いた曲線設計法である。このフレームワークを用いることで、クエリパターンやクエリの制約条件を考慮してクエリ性能を向上させることができることを、二種類の応用で 1 桁近く性能改善しうることを示した。

文　献

[1] M. Bader. *Space-Filling Curves: An Introduction with Applications in Scientific Computing*, volume 9 of *Texts in Computational Science and Engineering*. Springer Berlin Heidelberg, 2013.

[2] C. H. Hamilton and A. Rau-Chaplin. Compact hilbert indices: Space-filling curves for domains with unequal side lengths. *Information Processing Letters*, 105:155–163, 2008.

[3] D. Hilbert. Ueber stetige abbildung einer linie auf flächenstück. *Mathematische Annalen*, 38:459–460, 1891.

[4] S. Huang, B. Wang, J. Zhu, G. Wang, and G. Yu. R-hbase: A multi-dimensional indexing framework for cloud computing environment. In *Data Mining Workshop (ICDMW), 2014 IEEE International Conference on*, pages 569–574, Dec 2014.

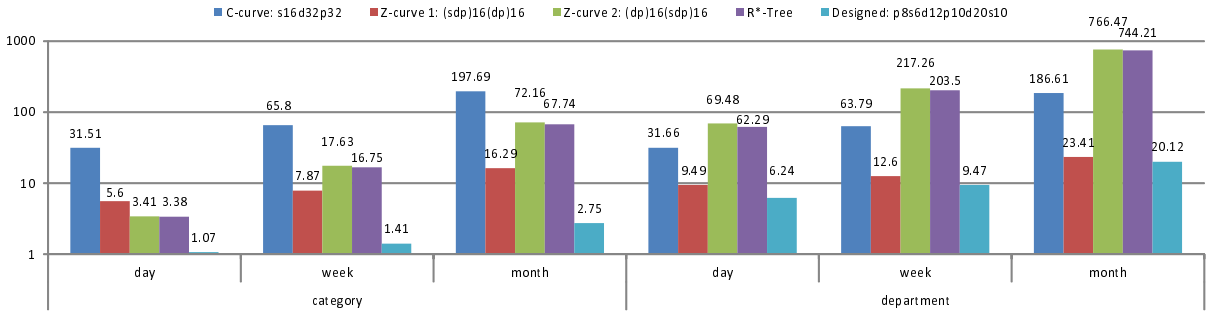


図 13　Average Number of Page Accesses by Retail Sales Analysis Query

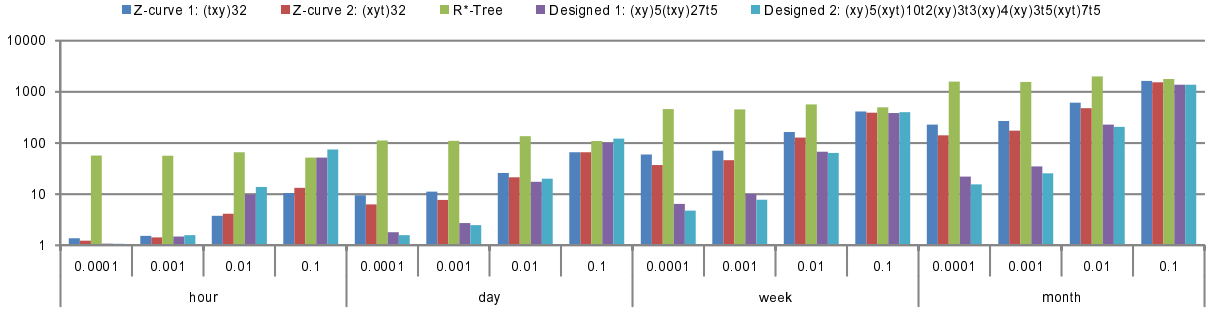


図 14　Average Numbers of Page Accesses through GIS Analysis Query

[5] R. Kimball and M. Ross. *The Data Warehouse Toolkit: the complete guide to dimensional modeling*. Wiley Computer Publishing, second edition, 2002.

[6] J. K. Lawder. Querying multi-dimensional data indexed using the hilbert space-filling curve. *SIGMOD Record*, 30:2001, 2001.

[7] V. Markl. *MISTRAL: Processing Relational Queries using a Multidimensional Access Technique*. PhD thesis, TU München, 1999.

[8] V. Markl and R. Bayer. Processing Relational OLAP Queries with UB-Trees and Multidimensional Hierarchical Clustering. *Proceedings of the International Workshop on Design and Management of Data Warehouses*, 2000:1–10, 2000.

[9] M. F. Mokbel and W. G. Aref. Irregularity in multidimensional space-filling curves with applications in multimedia databases. In *In Proceedings of the International Conference on Information and Knowledge Managemen, CIKM*, 2001.

[10] M. F. Mokbel, W. G. Aref, and I. Kamel. Analysis of multi-dimensional space-filling curves. *Geoinformatica*, 7(3):179–209, Sept. 2003.

[11] B. Moon, H. V. Jagadish, C. Faloutsos, and J. Salz. Analysis of the clustering properties of hilbert space-filling curve. *IEEE Trans. Knowl. Data Eng., TKDE*, 13(1):124–141, 2001.

[12] G. M. Morton. A computer oriented geodetic data base and a new technique in file sequencing. Technical report, IBM Ltd., 1966.

[13] S. Nishimura, S. Das, D. Agrawal, and A. El Abbadi. MD-HBase: A Scalable Multi-dimensional Data Infrastructure for Location Aware Services. In *Proceedings of 12th IEEE International Conference on Mobile Data Management, MDM*, pages 7–16, 2011.

[14] V. Raman, G. Attaluri, R. Barber, N. Chainani, D. Kalmuk, V. KulandaiSamy, J. Leenstra, S. Lightstone, S. Liu, G. M. Lohman, T. Malkemus, R. Mueller, I. Pandis, B. Schiefer, D. Sharpe, R. Sidle, A. Storm, and L. Zhang. Db2 with blu acceleration: So much more than just a column store. *Proc. VLDB Endow.*, 6(11):1080–1091, Aug. 2013.

[15] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[16] N. Shoji and Y. Haruo. Quilts: Multidimensional partitioning framework based on query-aware and skew-tolerant space-filling curves. In *Proceedings of the 2017 ACM SIGMOD International Conference on Management of Data*, SIGMOD '17, 2017. (to appear).

[17] T. Skopal, M. Krátký, J. s. Pokorný, and V. Snášel. A new range query algorithm for Universal B-trees. *Information Systems*, 31(6):489–511, Sept. 2006.

[18] L. Sun, M. J. Franklin, S. Krishnan, and R. S. Xin. Fine-grained partitioning for aggressive data skipping. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 1115–1126, New York, NY, USA, 2014. ACM.

[19] Sybase, Inc. *Performance and Tuning: Basics*, Aug. 2003. Chapter 13: Indexing for Performance.

[20] M. White. N-trees: large ordered indexes for multi-dimensional space. Technical report, Statistical Research Division, US Bureau of the Census, 1982.

[21] P. Xu and S. Tirthapura. On the optimality of clustering properties of space filling curves. In *Proceedings of the 31st Symposium on Principles of Database Systems*, PODS '12, pages 215–224, New York, NY, USA, 2012. ACM.

[22] Y. Zou, J. Liu, S. Wang, L. Zha, and Z. Xu. Ccindex: A complemental clustering index on distributed ordered tables for multi-dimensional range queries. *Network and Parallel Computing*, 6289:247–261, 2010.