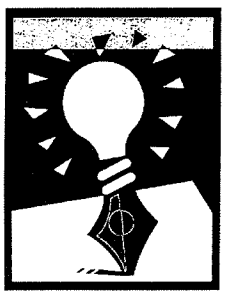
1 データストリームのためのマイニング技術



有村 博紀（北海道大学大学院情報科学研究科）
arim@ist.hokudai.ac.jp

喜田 拓也（北海道大学大学院情報科学研究科）
kida@ist.hokudai.ac.jp

大量の電子化データの流れであるデータストリームから，有用な情報を少ない資源で効率よく取り出すためのストリームマイニング技術を概観する．まず，データストリームの特徴と，データマイニングの目的について整理し，限定された計算資源を用いて無限に続くデータストリームからマイニングを行うためのシステムに要求される性質について議論する．次に，さまざまな要約データ構造とオンライン化技術についてまとめ，最近の研究のうち特色のあるものを紹介する．

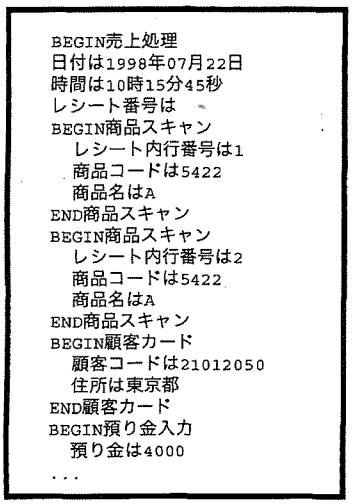
□データストリームからのデータ発掘

インターネットに代表される高速ネットワークと大規模センシング技術の発達によって，新しいタイプの大規模データである**データストリーム** (data streams) が注目を集めている[2]．

ここでデータストリームとは，時間的に変化する大量のデータレコードが日々刻々と生成・集積・消費される様子を，データの流れとしてとらえたものである．たとえば，金融や流通分野における取引記録 (transaction log)[2), 8)]や，電話会社・Webサービスプロバイダの通信記録 (call records/access log)[3)]などは典型的な例である．センサネットワークや，オンラインニュースや経済情報などの多数の情報源から生成されるデータもデータストリームと見なせる[2]．

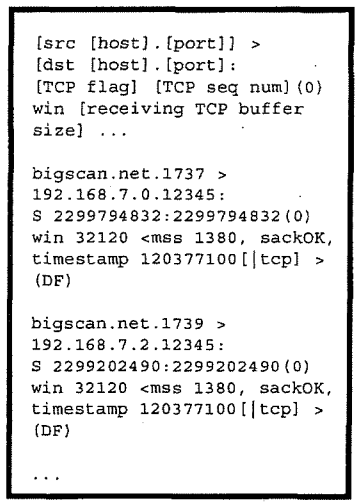
現在，通信分野や流通分野では，これらの大規模データストリームから，いつでも望むときに必要な情報を取り出したいという要求が高まっている．大規模データから有用な規則やパターンを見つける**データ発掘**または**データマイニング** (data mining) は，そのための有力な候補である．しかし，データストリームは (i) 膨大な量のデータが，(ii) 高速なストリームを通じて，(iii) 時間的に変化しながら，(iv) 終わりなく到着しつづけるという特性を持つ大規模データであり，従来の静的な大規模データベースを対象とした技術をそのままデータストリームに適用するには無理がある．したがって，限定された計算資源のもとで大規模データストリームを効率よく扱うための新しい情報処理技術が必要である．そこで，1990年代後半からデータストリームを対象としたデータ発掘技術，すなわち，**データストリームマイニング** (data stream mining) の研究が盛んになっている．

データストリームマイニングの研究は，1980年代以前の統計量の外部記憶計算や低メモリデータ構造の研究にはじまり，1990年代の通信分野でのストリームを対象とした統計処理システムの実証的研究や，計算量分野でのストリームアルゴリズムの理論的研究，データベース分野でのストリームに対する連続的問合せや集約計算



```
BEGIN売上処理
日付は1998年07月22日
時間は10時15分45秒
レシート番号は
BEGIN商品スキャン
  レシート内行番号は1
  商品コードは5422
  商品名はA
END商品スキャン
BEGIN商品スキャン
  レシート内行番号は2
  商品コードは5422
  商品名はA
END商品スキャン
BEGIN顧客カード
  顧客コードは21012050
  住所は東京都
END顧客カード
BEGIN預り金入力
  預り金は4000
...
```

(a)



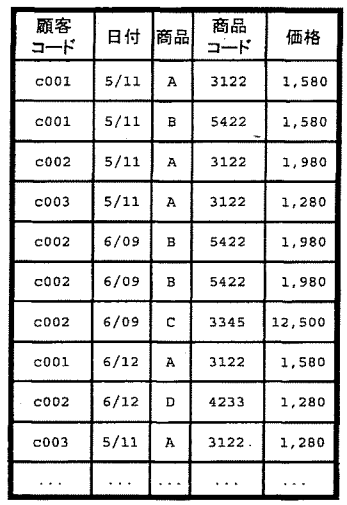
```
[src [host].[port]] >
[dst [host].[port]:
[TCP flag] [TCP seq num](0)
win [receiving TCP buffer
size] ...

bigscan.net.1737 >
192.168.7.0.12345:
S 2299794832:2299794832(0)
win 32120 <mss 1380, sackOK,
timestamp 120377100[|tcp] >
(DF)

bigscan.net.1739 >
192.168.7.2.12345:
S 2299202490:2299202490(0)
win 32120 <mss 1380, sackOK,
timestamp 120377100[|tcp] >
(DF)

...
```

(b)



顧客コード	日付	商品	商品コード	価格
c001	5/11	A	3122	1,580
c001	5/11	B	5422	1,580
c002	5/11	A	3122	1,980
c003	5/11	A	3122	1,280
c002	6/09	B	5422	1,980
c002	6/09	B	5422	1,980
c002	6/09	C	3345	12,500
c001	6/12	A	3122	1,580
c002	6/12	D	4233	1,280
c003	5/11	A	3122	1,280
...	...	...	...	...

(c)

図-1 典型的なデータストリームの例

の研究などを経て，次第に注目を集めるようになった．2000年代に入って顕在化し，現在，データストリームを対象としたデータ解析・機械学習技術の研究が盛んに行われている．また，情報検索分野や時系列モデリング分野でも，関連した研究が精力的に行われている．

本来，データ発掘は，基幹業務系システムにおいて，日常的なトランザクション処理の履歴データを，データ倉庫 (data warehouse) に集約・格納し，意思決定支援に有効利用することからはじまったといわれている[8), 12)]．この意味で，データストリームは古くて，かつ新しい対象であるといえる．

データストリームを対象としたデータ発掘技術は，現在も発展中の新しい情報技術である．本稿では，このデータストリームマイニングについて，それが何であるか，現在までに何が分かっているか，また，それらの基本的なアイディアは何かといった点に焦点を当てながら，これまでに提案されたさまざまな特色ある技術を主にアルゴリズムの視点から紹介する．

□大規模データストリームとは？

データストリームは，トランザクションデータの連続した流れである．典型的なデータストリームの例を**図-1**に示す．

図-1の左 (a) は，購買情報データの例である．大規模小売店では，このような顧客購買情報が，POS (point of sales) 情報システムを通じてオンラインで収集され，業務処理以外にも販売最適化や商品開発などに利用されている[8)]．たとえば，米国のウォルマートでは2003年時点で1日あたり1億人分前後の購買情報が，米国全土の4,400の店舗から収集されており，その総量は104週分で約10万品目70テラバイトに上るといわれている．

図の中 (b) は，通信記録データの例である．計算機ネットワークでは，ネットワーク解析ツールによって生成されたこの種のパケット解析記録が，稼働状況の監視や不正侵入検出の目的で用いられる．たとえば，10Gbps前後の容量を持つ基幹線の場合，これらの解析記録は10～100GB/時と多量である．また，電話回線の通信記録も典型的なストリームデータである．CortesとPregibonら[3)]は，AT&Tが1990年代後半に行った8千万レコード／日と3億レコード／日の携帯電話と長距離電話の通信記録 (call record) を対象としたデータマイニング事例について報告している．

それでは，これらのデータストリームを対象として，データストリームマイニングではどのような問題を研究してきたのだろうか？ 1つは，属性値に関する平均やかたより，値の総和といった簡単な統計量を，きわめて小さなメモリと処理時間だけを使って計算する方法の研究である．もう1つは，結合規則や，決定木，クラスタリングといった機械学習やデータマイニングにおける規則やパターンの発見アルゴリズムを連続したデータストリームへ拡張する研究である．

具体的に上の顧客購買情報データを例にとって説明すると，これまでに，次のようなデータマイニング問題を効率よく解くアルゴリズムが研究されてきた．

- 過去に購買された商品の種類数 (**異なり数**)[5)]
- 商品ごとの購買数のかたより (**分散**)[6)]

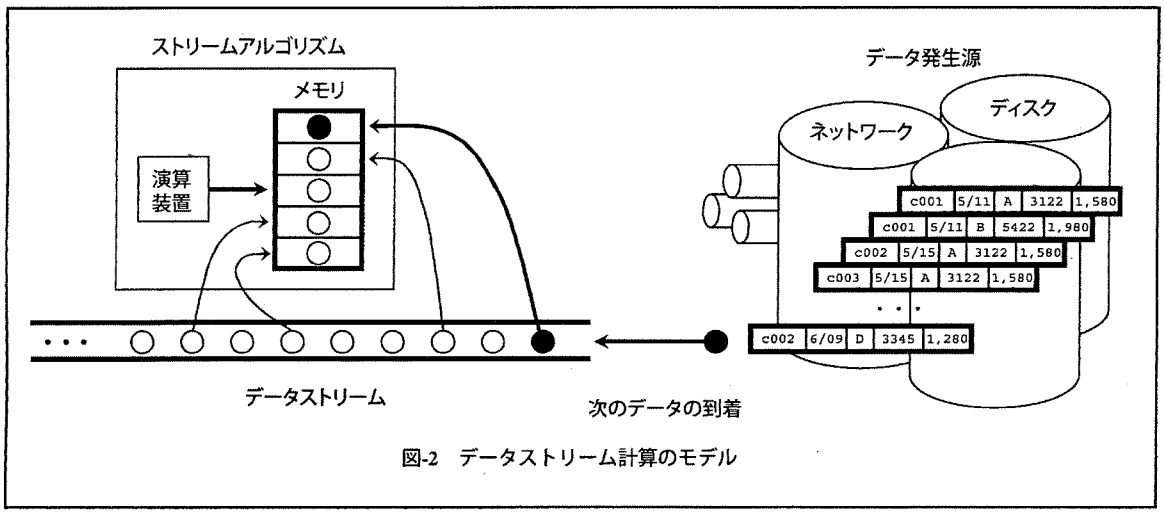


図-2 データストリーム計算のモデル

- 過去の一定期間内の売上げ総和(**滑り窓和**)[2), 6)]
- 一定数以上売れた商品のリスト(**頻出値**)[10)]
- 売り上げ最上位K個の商品(**ホットリスト**)[6)]
- 同時によく買われる商品の**組合せパターン**[10)]
- 購買傾向を予測するための**分類規則**[4)]
- よく似た商品の**クラスタリング**[11)]

現在までの研究で，これらの問題に対して，小さなメモリしか使わない1パス統計量計算アルゴリズムや，膨大なストリームに対して適応的に働くさまざまなオンラインパターン発見手法が提案されてきている．次章以降では，これらの研究を紹介する．

□データストリーム計算のモデル

ここで，アルゴリズムの効率に関して正確な議論をするために，前章で見たような現実のデータストリームを抽象化して，**図-2**のようなデータストリーム計算のモデルを導入する[6)]．

データストリームは，図-1の右(c)に示したような，一連の属性(顧客コードや日付，商品コード等)に対する値の組合せである**レコード**の系列である．図-2に示すように，**計算機**は有限の**メモリ**と計算能力を持ち，データストリームからレコードが次々とオンラインで到着するたびにメモリ上で計算を行い，ときどきは利用者からの要求に応じて**答え**を報告する．

アルゴリズムの効率は，使用するメモリ量の最大値とデータ1個当たりの内部データの更新時間で測ることが多い．有限ストリームモデルの場合は，データのアクセスは逐次走査に限定し，パス数(データの走査数)や入出力回数も考慮する．無限ストリームモデルの場合は，データを再読み込みできないので，1パス処理が必須である．

データストリームを対象としたアルゴリズム研究の目標は，膨大で高速なデータストリームに対して

- 非常に少ない計算資源を用いて，
- 長期間安定して動き続け，
- 利用者の要求に即応して，
- 近似的な解を返す

ような特性を持つアルゴリズムを設計することである[2), 6)]．

□アルゴリズムの基本技法

データストリームマイニングでは，無限のデータストリームに対して，有限の計算資源だけを用いて，効率よい計算を行う点に本質的な難しさがある．そのため多くのストリームマイニング手法では，厳密解を計算することはあきらめて，近似的な解や確率的な解を求める戦略をとることが多い．また，**データ要約**(synopsisやsketch)[6)]と呼ばれる小さなデータ構造を主記憶上に保持することで，効率よい計算を実現する戦略をとっている．

ここで，データストリームを対象としたマイニングアルゴリズムを構築するための基本的な技法をまとめておこう．大きく分けて，これには次の3つがある．

1つ目は，**確率的計算**である．厳密解の決定的計算をあきらめて確率的計算を用いることで，単純で効率のよいアルゴリズムが得られることも多い．たとえば，後でみる近似計数のように，整数カウンタを用いずに確率的に数を数えることができる．2つ目は，**データの粗視化**である．詳細な情報を得るためにすべてのデータをメモリに保持するのでなく，データをグループ化しておおまかな情報(グループの代表元や統計量)を保持することで，データを圧縮してメモリを節約する戦略である．3つ目は，**適応的計算**である．最終的な解が分かっていれば必要なデータだけを主記憶に保持すればよいような

問題も多い．そこで，最初は粗い解からはじめて，計算が進んで情報が増えてきたら，徐々にデータ管理を詳細化していく戦略である．

実際には，上記の3つの手法を互いに組み合わせて用いることが多い．また，どの手法を採用しても，必然的に近似アルゴリズムになるのが普通である．

□頻出値とホットリスト

それでは，膨大なデータストリームに対して小さな主記憶しか使わないような効率よい計算が可能なのだろうか．この疑問に答えるための例として，「一定数以上売れた売れ筋商品のリスト」を求める問題，すなわち頻出値問題を有限の主記憶だけを用いて解く方法を文献9), 10) に従って紹介しよう．

ここで**頻出値問題**とは，与えられた正数$0 < \sigma \leq 1$ (相対頻度と呼ぶ) に対して，N個の値からなるデータストリームA中に$N\sigma$回以上の頻度で出現する値 (**頻出値**と呼ぶ) をすべて見つける問題である．面白いことに，答えの総数は入力サイズNに関係なく，たかだか$\delta = 1/\sigma$個である☆1．そこで，究極の目標は答えのサイズ$O(\delta)$個程度の領域だけを用いて頻出値を計算することである．

実感が湧くように，今，部長さんがたくさんの営業マンを部下に持っていて，1年間のN回の契約のうち$N\sigma$回以上の契約をとった成績優秀者を表彰しようとしていると考えよう．部長さんは部下が契約の報告にくるたびに，手帳の1行に1人分ずつ名前と契約数を記録して，成績優秀者をもれなく見つけようとする．ここで，最初から有望そうな部下が分かっていれば，彼らの成績だけを手帳に記録すればよいのだが，手帳に記録しない限りは有望かどうかは分からない．しかし，全員の成績を記録したのでは，手帳があふれてしまうというジレンマが本質的である．出てきたら記録するという単純な方式では，最悪時に値の総異なり数程度のメモリ量が必要になる．

そこで，最初はとりあえず適当にはじめておき，その時点までの実績に応じて手帳の管理を行う**適応的**戦略をとることにする．ここで，成績優秀者は年間に少なくとも$N\sigma$個の契約をとるから，δ回 ($= 1/\sigma$回) に1回は彼が契約をとるはずだと推論すると，とりあえず次のような

☆1 各頻出値は$N\sigma$回以上出現するから，$N/N\sigma = 1/\sigma = \delta$となる．この値$\delta$はこの問題で繰り返し現れるパラメータである．

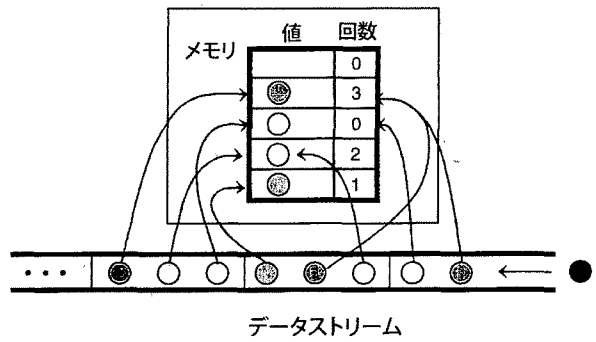


図-3 頻出値のマイニング

方式が考えられる．

頻出値の計算「ノルマ方式」

- **図-3**のように，メモリ上に値をキーとするハッシュ表 (手帳) をとる．
- 部下が契約の報告にくるたびに次を行う．手帳に名前が載っていなかったら，契約数の初期値を1として無条件で手帳に載せる．すでに載っていたら，契約数を1つ増やす．
- 次に，δ回からなる期間ごとにハッシュ表上の候補者全員に対して，期間内に少なくとも1回は契約をとるというノルマを達成したか調べる．そして，ノルマを達成できなかったものを手帳から除く．

残念ながら，先の推論の後半は間違っており，ノルマの適用が厳しすぎるため，本来の成績優秀者であっても，スランプでタイミング悪く契約数ゼロの週が続くと手帳から消えてしまう．そのためノルマ方式では頻出値の取りこぼしが起きる．そこで，MankuとMotwani[10]は，このノルマ法を次のように修正することで，この問題を回避できることを示した．

頻出値の計算「改良ノルマ方式」

- 部下が最初に報告にきたのがt番目の期間だったら，(以前に運悪くスランプで落伍しているかもしれないので) 実は手帳にない期間すべてで目標を達成していると解釈して，契約数の初期値を1ではなくてtから始めてやる．
- その他は改良ノルマ方式と同じように計算する．

この改良方式では成績を過小評価することはあり得ないので，すべての成績優秀者が見つかる．驚くべきことに，このかなり温情的な改良ノルマ方式でも，最適値からわずかに$\log_e N$個分多いだけの箱$O(\frac{1}{\sigma}\log_e N)$個のメモリ量しか使用しないことが示せる[10]．また契約数の現在の近似値fとともに初期値のtをハッシュ表に記録

しておけば，真の契約数f_*は$f-t \leq f_* \leq f$を満たす範囲にあることもいえる．

さらに，Karp, Shenker, Papadimitriouら[9]は，次の連帯責任方式によって，すべての成績優秀者を出力に含みつつ，極限の箱$O(\delta)$個分まで使用領域を節約することが可能なことを示した．

頻出値の計算「連帯責任方式」

- 図-3のように，メモリ上に値をキーとし，ちょうど$\delta+1$個の箱☆2からなるハッシュ表（手帳）をとる．
- 部下が契約の報告にきたとき，手帳に名前が載っていなかったら，契約数の初期値を1として無条件で手帳に載せる．
- もし新しい部下がきたときに手帳がいっぱいになっていたら，いったん記録を中断して，契約数が0回の者がでるまで，全員の記録から1回分ずつ契約数を引くことを繰り返す．最終的に契約数が0回のものが出たら，それらの名前をすべて除く．新しい部下の名前をハッシュ表に記録し，再開する．

上記のいずれの方式も，事前にNを知る必要はなく，無限データストリームに対してもそのまま働く．上でみたように「早すぎる判断を避けること」や「近似解で満足すること」によって，頻出値問題をほぼ最適に近い$O(\delta)=O(\frac{1}{\sigma})$個分の領域量で解くことができた．上記の手法は，データストリームからの頻出アイテム集合発見に応用されている[10]．関連する研究として，筑波大の吉田らによる，直接マップキャッシュ（direct mapped cache）のアイディアを用いたきわめて高速なスパムメール検出手法の提案がある[13]．

□統計量の近似計数

データマイニングの基本は，数を数えることである．ここでは，基本技法の1つである**確率的計算**によって，データストリーム上でさまざまな統計量を定数メモリ量で計算する方法を説明しよう．

今，長さNのデータストリームAに出現する値aについて，その出現数を$f_A(a)$とおくと，非負整数k=0, 1, 2 …に対して，**k次の頻度モーメント**は

$$F_k=\sum_{a\in A}(F_A(a))^k$$

と定義される☆3．F_0はAに出現する値の異なり数であり，F_1は総出現数，F_2はかたより，F_∞は最大値を表す．これらは簡単な統計量だが，無限ストリームに対して，定数メモリ量で計算するのは容易ではない．

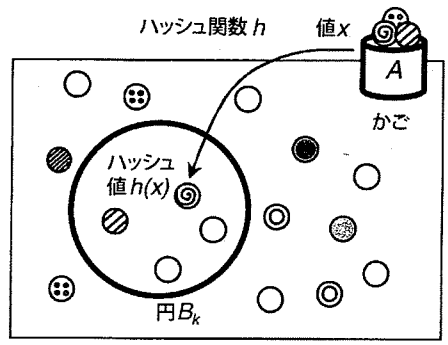


図-4 異なり値F_0の確率的計算

異なり数F_0を確率的に数える

はじめに，1983年にFlajoletら[5]によって提案された値の異なり数F_0を**確率的**に数える方法を与えよう．異なり数の計算の難しさは，一度見た値を覚えておく必要がある点である．次に示すアルゴリズムは，確率的なアイディアを用いて，値を記憶することなく，異なり数を定数個のレジスタ（すなわち$O(\log N)$ビット）で近似的に求める．

基本的なアイディアを説明するために，一種の玉入れゲームを考えよう．**図-4**に示すように床の面積が1の体育館に，面積が$0<\alpha<1$の円が描いてあるとする．

異なり値の計算「玉入れゲーム方式」

今，かごAの中に色とりどりのボールが入っていて，体育館の2階から広い床の上にそれらのボールを投げ込む．ここで，一種のハッシュ関数hによって，同じ色のボールは床の上の決まった位置に着地するとする．床の上には，円Bが描いてあり，そばには小人が1人待機している．すべてのボールを投げ込み終わったら，小人は円Bの中に入った異なる色のボールの数Cを数えて報告する．最後に，異なり数の推定値を$M\simeq C/\alpha$とする．

この話では，かごAがデータストリームであり，ボールの色はそれぞれの値を表している．重要な仮定として，ハッシュ関数hは，玉入れゲームをするごとにランダム

☆2 オリジナル[9]では頻度が$1/\sigma$より真に大きい値をδ個の箱を用いて発見するが，ここでは文献10）に合わせた．
☆3 正確には全体を$\frac{1}{k}$乗するが，ここでは慣習により省略する．

に選び，さらに，投げ込んだボールが床の上にうまく散らばるように，各値xに対して$h(x)$が値域上で一様分布するとする．

上の仮定より，かごの中にM個の異なる色が含まれているならば，円Bが含む異なる色を持つボールの数Cはその面積に比例するはずである．実際，床全体の面積に対する円の面積の比率を$0 < \alpha < 1$とおくと，異なる色のボールの期待値は$E[C] = \alpha M$となることがいえる．逆に$E[C]$としてCの平均値を用い，$M \simeq E[C]/\alpha$を求めることができる．

しかし，このままでは，依然として小人は異なる色のボールの数を数える必要がある．そこで，次のように複数の円を使って，円1つあたり1ビットでおおまかに数えることにする．

異なり値の計算「改良玉入れゲーム方式」

今度は，十分大きな数$d \geq \lceil \log M \rceil$に対して，面積が$1, 1/2, 1/4, \ldots, 1/2^d$と$1/2$ずつ小さくなるように，いくつかの円$B_0, B_1, B_2 \ldots, B_d$を床の上に描く．これまでと同様に，かご$A$のボールをすべて投げ込んでから，それぞれの円の中にボールが入っているかだけを調べる．最後に，ボールを含む最小の円$B_{k_{\min}}$を見つけて，異なり数の推定値を$M \simeq 1/\alpha_{k_{\min}}$とする．

円B_kの床全体に対する面積比は$\alpha_k = 1/2^k$となる．統計的なゆらぎの解析☆4から，新しい方式では$\alpha_k \simeq 1/M$程度の場合に，高い確率で円B_kにはおよそ1個のボールが含まれ，大きさが半分の円B_{k+1}には0個のボールしか含まれないことを示せる．

ビット演算を用いて，2^d以下の二進数全体で床全体を表し，最上位ビットが$d-k$の二進数全体で円B_kを表す．さらに，落下したボールを含む最小の円の添え字を1ビットで記録する．すると，1個の整数レジスタと，ハッシュ関数のための定数個のレジスタを用いて，この方式を実装できる[5]．

この方法は二進数で2ビット程度の誤差を出す確率が約1/2と，乱暴なやり方の割にはそう悪くない．改善案として，上のアルゴリズムを複数個並列に走らせ，出力の平均をとる手法が提案されている[5]．

☆4 Markovの不等式とChevyshevの不等式を用いて，望ましい状態が起きない確率を$O(1/c)$で抑えられる[6]．

☆5 元のアルゴリズム名はTugOfWarで，綱引きの意．

かたよりF_2を確率的に求める

今度は値のかたよりF_2を，定数個のレジスタを用いて確率的に計算する方法[6]を説明する．一度見た値を覚えておけない点は，先のF_0の計算と同じである．

このアルゴリズム「総当たり戦」☆5の基本的なアイディアは，**ランダム射影**と呼ばれる，要素に正負の符号を割り当てるハッシュ関数$h : U \to \{+1, -1\}$を用いることである．ここでUは値の全体である．

かたよりの計算「総当たり戦方式」

はじめに，N人の若者たち$A = (a_1, a_2, \ldots, a_N)$を運動場に並べる．次にランダム射影hを1つ選び，個々の若者にランダム射影をかけて，正負の値の影武者を作り，$h(a_1), h(a_2), \ldots, h(a_N) \in \{-1, +1\}$と1列に並べて，これを紅組とする．同様に，影武者の列のコピーをもう1つ作って白組とし，号令をかけて紅組全員と白組全員を総当たり戦で対戦させる．すると，それぞれの影武者の積$h(a_i) \cdot h(a_j) \in \{-1, +1\}$がたくさんできるので，これらを足し合わせて和$X \in \boldsymbol{Z}$を求める．以上の総当たり戦を，ランダム射影を取り替えては何回も繰り返して，和Xの平均値を求め，これをF_2の推定値とする．

ここで，ランダム射影はランダムなhの選択と，任意の値$x, y \in U$に対して，次の2つの性質 (i) $h(x)$の期待値は$E[h(x)] = 0$であることと，(ii) $h(x)$と$h(y)$は独立であることを満足する[6]．

総当たり戦の結果Xを求めるには，影武者の列を$Z = h(a_1) + h(a_2) + \ldots + h(a_m)$と線形和で表し，この二乗$X = Z^2$を計算する．ストリームに対して，これを実現するには，整数レジスタを$Z = 0$で初期化し，要素a_iを読むたびにハッシュ値$h(a_i)$をZに足しこんでいく．すべて読み終わったら，正整数$X = Z^2 = \sum_{0 \leq i,j \leq m} h(a_i), h(a_j)$を計算すればよい．これで，総当たりでの和$X$が実現できる．

ここで，異なるランダム射影について影武者の積の期待値をとると，同じ値同士の積は$E[h(a), h(a)] = 1$となり，違う値同士$a \neq b$の積は$E[h(a), h(b)] = 0$となるので，$X = Z^2$の期待値は次のようにF_2になる．

$$E[Z^2] = \sum_{1 \leq i,j \leq m} E[h_s(a_i)\, h_s(a_j)]$$

$$= \sum_a f(a)^2 \cdot 1 + \sum_{a \neq b} f(a)\, f(b) \cdot 0 = F_2$$

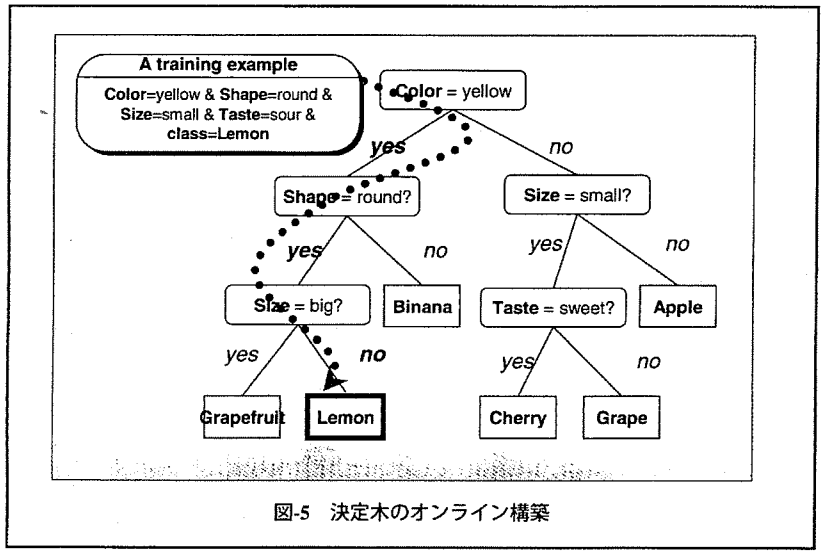


図-5 決定木のオンライン構築

さらに，解の分散を抑えるために，複数の手続きを並列にストリーム上で走らせて，答えの中央値を出力する．すると，ランダム射影がさらに4独立性☆6を持つとき，高い確率で与えられた誤差以内でF_2が求まる[6]．実際に，Alonらは，データベースの結合演算サイズの推定にこれを適用し，従来法より少ない$C = 64 \sim 256$個程度のレジスタで良好なF_2値が計算できたと報告している．

□決定木のオンライン構築

決定木構築は，古くから機械学習や統計の分野で研究されてきた代表的なデータマイニング問題である．

決定木(decision tree)は，**図-5**に示すような木構造をした予測規則であり，与えられた関係レコードに対して，根から出発して，レコードの属性値に関する質問に答えながら各節点を訪問し，最終的にたどり着いた葉の持つ値を予測値として出力する．

C4.5等に代表されるオフライン型決定木構築方式は，すべての訓練データを入力として受け取り，統計的に最もよくデータを分割する属性と質問を選びながらデータを再帰的に分割し，同時に，決定木を再帰的に構築する．しかし，この方式では全データがそろわないと構築を開始できない上に，データへのランダムアクセスが必要なので，データストリームには適用できない．

代表的なオンライン型決定木構築方式であるVFDT (Very Fast Decision Tree Learner)[4]では，決定木を根からはじめて次第に成長させていくところまではC4.5と同じだが，全データの到着を待たずに**適応的**に木を生長させていく．VFDTは，新しいデータを受け取るごとに，現在の木でそれが到達する葉を求め，そこに到着したデータを蓄える．すると，よく現れるタイプのデータが特定の葉に貯まってくる．適当な統計的基準☆7でを満たすほどデータが貯まったら，より詳しい予測を行うために貯まったデータを使ってその葉をさらに成長させる戦略を採用している．

このような適応的戦略は，複雑な構造を持つパターンのマイニングに有効である．文献7)は，適用的戦略をデータストリームからの結合規則発見に適用し，文献1)は，半構造データストリームからの頻出構造発見に適用している．本特集の山西らの解説にも，適用的学習アルゴリズムが取り上げられている．また，PPM等の可変長文脈を用いた適応的圧縮アルゴリズムも関連が深い．

□ストリーム指向のクラスタリング

もう1つの代表的なデータマイニング問題が，クラスタリングである．

クラスタリング(clustering)は，適当な距離尺度のもとで，データ集合を互いに似通ったデータからなるいくつかのかたまり(**クラスタ**と呼ばれる)に分割することである．階層的クラスタリングや，k-Meanクラスタリングなどのオフラインのクラスタリングの多くは，すべてのデータどうしを比較するため，データ数Nに対して$O(N^2)$の計算時間とデータへのランダムアクセスを要する．したがって，実時間かつ1パスでのデータストリー

☆6 (iii) hが4独立(4-wise independent)であるとは，任意の異なる値x, y, z, wに対して，ベクトル$(h(x), h(y), h(z), h(w))$の16通りの実現値の確率がそれぞれ1/16であることをいう．

☆7 文献4)ではChernoff/Hoeffding不等式を用いている．

ム計算は難しい．

この問題に対して，クラスタリング手法BIRCH[11]はデータを**粗視化**することでクラスタリングの大規模化に成功している．基本的なアイディアは，クラスタを1つの点として粗視化してデータを圧縮し，データ数と使用領域の両者を減らす点である(**図-6**)．たとえば，代表点の数を$M = \sqrt{N}$個に減らすことで，データ点を主記憶におさめることができると同時に，通常の二乗時間アルゴリズムを用いて$O(M^2) = O(N)$時間でクラスタリングが可能になる．

BIRCHは，はじめにデータストリームを走査しながら，定められた半径以下の小さなクラスタをオンラインで生成する．これは，B木に似た要約データ構造を用いて，主記憶上で管理されたクラスタ群にデータを挿入していくことで行われる．生成されたクラスタは，クラスタ中心を代表点として，クラスタ内のデータの個数や，平均，かたより等の統計量とともに粗視化される[11]．BIRCHはストリーム走査の終了後に，代表点とともに記録された統計量を利用して，もう一度主記憶上で階層クラスタリングやk-Meanなどのクラスタリングを用いて，最終的なクラスタを生成する．

BIRCHの粗視化のアイディアは理論的にも有用であり，最近GuhaとMeyerson，Mishra，Motwaniらは，階層的な粗視化を用いて，k-Medianクラスタリング問題を1パスで解く定数近似率アルゴリズムを与えている．

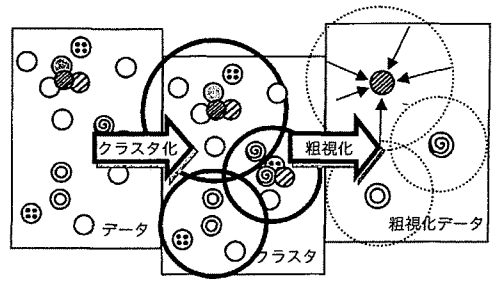


図-6　ストリーム指向のクラスタリング

□おわりに

本稿では，最近のデータストリームを対象としたデータマイニングについて，基本的な統計量計算から機械学習アルゴリズムのオンライン化まで，最近の結果を紹介した．

本稿では触れることができなかったが，時系列ストリームや分散環境でのセンサストリームからのデータ発掘，ネットワークセキュリティへの応用は，最近注目されている話題である．ご興味をお持ちの方は，本特集の山西氏らの解説や，文献2)，6) などをご覧いただきたい．データストリーム処理全般については文献2) が，要約データ構造については文献6) が，クラスタリングについては福田，森本，徳山らの教科書が良い解説である．

謝辞　喜連川優先生，北川博之先生，石川佳治先生をはじめとする特定領域研究「情報学」のA02柱「コンテンツの生産・活用に関する研究」メンバ各位にはデータストリーム処理に関して貴重なご教示とコメントをいただきました．また，九州大学情報基盤センター研究部の笠原義晃先生には，ネットワーク管理に関するご教示をいただきました．ここに感謝いたします．本稿の一部は浅井達哉君，安部賢治君，川副真治君，河野正太郎君他の旧研究室メンバとの議論と共同研究によるものです．謝意を表します．

参考文献

1) Asai, T., Arimura, H., Abe, K., Kawasoe, S. and Arikawa, S.: Online Algorithms for Mining Semi-structured Data Stream, in Proc. IEEE ICDM'02, pp.27–34 (2002).
2) Babcock, B., Babu, S., Datar, M., Motwani, R. and Widom, J.: Models and Issues in Data Stream Systems, in Proc. ACM PODS'02, pp.1–16, ACM (2002).
3) Cortes, C. and Pregibon, D.: Signature-based Methods for Data Streams, Data Mining and Knowledge Discovery, 5:167–182 (2001).
4) Domingos, P. and Hulten, G.: Mining High-speed Data Streams, in Proc. ACM SIGKDD'00, pp.71–80 (2000).
5) Flajolet, P. and Martin, G. N.: Probabilistic Counting, in Proc. FOCS'83, pp.76–82 (1983).
6) Gibbons, P. B. and Matias, Y.: Synopsis Data Structures for Massive Data Sets, in J. Abello, editor, External Memory Algorithms, Vol.50 of DIMACS Series in Discrete Mathematics, pp.39–70, DIMACS (1999).
7) Hidber, C.: Online Association Rule Mining, in Proc. ACM SIGMOD'99, pp.145–156 (1999).
8) Inmon, W. H.: Building the Data Warehouse, Wiley, 3rd edition (2002).
9) Karp, R. M., Shenker, S. and Papadimitriou, C. H.: A Simple Algorithm for Finding Frequent Elements in Streams and Bags, ACM Trans. Database Syst., 28:51–55 (2003).
10) Manku, G. S. and Motwani, R.: Approximate Frequency Counts over Data Streams, in Proc. VLDB'02, pp.346–357 (2002).
11) Zhang, T., Ramakrishnan, R. and Livny, M.: Birch: A New Data Clustering Algorithm and its Applications, Data Min. Knowl. Discov., 1(2):141–182 (1997).
12) 喜連川優：データベースとデータマイニングにおける並列処理，特集「計算機クラスタ」，情報処理，Vol.39, No.11, pp.1089–1094 (Nov. 1998).
13) Yoshida, K., Adachi, F., Washio, T., Motoda, H., Homma, T., Nakashima, A., Fujikawa, H. and Yamazaki, K.: Density-based Spam Detector, in Proc. ACM SIGKDD'04, pp.486–493 (2004).

(平成16年12月7日受付)