線型予測ソート

１．概要

ランダムなデータがソートされた後、その値がほぼ直線上に並ぶと仮定します。

昇順にソートされたデータについて、ｘ軸にデータの順番（ｘ＝１～ｎ）をとり、ｙ軸にデータの値をとった平面を考えます。このデータの集合（x_i、y_i）（ｉ＝１～ｎ）が一次式　ｙ＝ａ　ｘ　＋ｂ　を満たすとします。

ここで元のランダムなデータ　y_0を１次式に代入して根を求めます。この根が求めるデータの順番になります。

２．線型予測ソートのアルゴリズム

このソートでは１次式で計算して求めた値を記録しておくために別の領域を必要とします。いわゆる外部ソートと呼ばれるものです。

（１）データの集合（x_i、y_i）（ｉ＝１～ｎ）全体を調べて、最小値（ｘ min、ｙ min）と最大値（ｘ max、ｙ max）を求め、１次方程式を決定します。

$$y = \frac{(y\ max - y\ min)}{(x\ max - x\ min)} + y\ min$$

（２）データの集合を調べ、１次方程式からその順番を決めて記録する。データ　（x_i、y_i）の順番がｊと計算されたときは、これを（ｘｊ、y_i）として記録します。

ただし、同じ順番のデータが複数ある場合にはリンクさせておきます。

（３）新しく記録したデータの集合は空きやリンクがあるので、これを修正して元の集合に戻します。

（４）配列を使った例

１次式は　y=(88-4)x/(99-1)+4=6x/7+4　で、　x=7(y-4)/6 となります。

						min			max						
a［x］	1	2	3	…	…	23	…	…	51	…	…	77	…	…	99
	16		33			5			4			88			71
	↓		↓			↓			↓			↓			↓
	x=16.3		33.8			1.2			0			98			78.2

b［x］	1	2	…	…	16	…	…	33	…	…	78	…	…	98	99
	5				16			33			71			88	
	4														

３．線型予測ソートの評価

探索の手間はO(kN)ですが、リンクデータをソートしますので、その手間がO(m log m)になり、合計するとO(N+m log m)になります。

理論的にはすべてのデータが異なるときがリンクがないので最も速く、すべてのデータが同じときには全部リンクされるので最もなるはずです。

ＬＰＳｏｒｔ．ｅｘｅというサンプルプログラムを作成して、データの個数（1 万～ 100 万個）とデータの性質（ランダム、正規分布、逆順）を変えて調べてみたところ、おおむねクイックソート＞線型予測ソート＞コムソート＞基数ソートの順で速くなりました。

やはりクイックソートは最強のソートと言われるだけに最も速くなりましたが、線型予測ソートも健闘しました。

ただし、動的配列を使うとその確保のために時間が掛かるのですべて静的配列を使いました。しかし現実のプログラムでは動的配列を使うことになり、そのときはクイックソートの数倍～１０倍の時間が掛かってしまいます。

サンプルプログラムは VisualBasic5.0 コードを書いて実行速度を最適化するネィティブコードコンパイルをしましたが、P-Code コンパイルをしたとき速度は線型予測ソート＞クイックソートになりました。理由は不明です。

基数ソートはO(N)に比例するはずなのに意外に遅い結果になりました。また線型予測ソートは基数ソートと異なり実数データのソートもできます。

３．線型予測ソートのソースコード

```
Private Type Tlink
    link(20) As Double
End Type

Const nsize = 1000000

Dim data(nsize) As Double       元データ
Dim tmp(nsize) As Tlink         作業領域
Dim c(nsize) As Long            リンク情報

Private Sub LPSort()
Dim i As Long
Dim j As Long
Dim m As Long
Dim k As Long
Dim l As Long
Dim p As Long
```

```
Dim q As Long
Dim dmax As Double          データの最大値
Dim dmin As Double          データの最小値
Dim a As Double
Dim b As Double
Dim gap As Long
Dim sw As Long
Dim dummy As Double

    dmax = data(1)
    dmin = data(1)
    For i = 1 To ndata
        If dmax < data(i) Then dmax = data(i)
        If dmin > data(i) Then dmin = data(i)
    Next i

    a = (ndata - 1) / (dmax - dmin)                    一次式
    b = -(ndata * dmin - dmax) / (dmax - dmin)

    For i = 1 To ndata
        x = a * data(i) + b
        m = Int(x)                          データの場所
        If m <= 0 Then m = 1
        If m > ndata Then m = ndata
        '重複するかの判定
        If c(m) = 0 Then
            tmp(m).link(1) = data(i)
        ElseIf c(m) > 0 Then
            tmp(m).link(c(m) + 1) = data(i)          リンクさせる
        End If
        c(m) = c(m) + 1
    Next i

    k = 0
    For i = 1 To ndata
        If c(i) = 0 Then
        ElseIf c(i) = 1 Then
            'データが 1 個
            k = k + 1
            data(k) = tmp(i).link(1)
```

```
        ElseIf c(i) > 1 Then
            'データが重複
            l = c(i)
            'コムソート
            gap = l  'データの個数
            Do
                gap = Int(gap / 1.3)
                If gap < 1 Then
                    gap = 1
                End If
                sw = 0
                For p = 1 To l - gap
                    q = p + gap
                    If tmp(i).link(p) > tmp(i).link(q) Then   'データ交換
                        dummy = tmp(i).link(p)
                        tmp(i).link(p) = tmp(i).link(q)
                        tmp(i).link(q) = dummy
                        sw = sw + 1
                    End If
                Next p
            Loop Until((sw = 0) And (gap = 1))
            'コピー
            For j = 1 To l
                k = k + 1
                data(k) = tmp(i).link(j)
            Next j
        End If
    Next i

End Sub
```