適応整列法の評価

4J－10　二村 良彦　遠藤 貢一　平井 利治　青木 健一
早稲田大学 理工学部

１．はじめに

整列済みに近い数列を高速に整列する，いわゆる適応整列法(adaptive sorting algorithm)が近年注目されている[3]．本研究は，LOAS[5]，MELソート[9]，SKIPソート[9]等の比較的新しい適応整列法の性能を，QUICKソートおよびMERGEソート等の実用的整列法と比較するものである．数列の整列度を系統的に変えながら整列法の性能(キーの比較回数，実行時間等)の変化を調べる実験は既にいくつか行われている[1,2]．本稿では，葉数(数列において隣人よりも値の小さな要素の個数)[5]と呼ばれる新しい整列性測度に基づき各種整列法を評価する．整列性測度としての葉数の妥当性および葉数を制御して順列をランダムに生成する方法は各々文献[5]および[4]で報告した．本研究によって得られた結論は次の2点である：(1)キーサイズが小さいときは，葉数とほぼ無関係にQUICKソートが一番速い．(2)キーサイズが大きい時は葉数が大きくてもLOAS等の適応整列法が速い．

２．適応整列法

我々はさきに，自然マージソート[7]およびその改造型のRUNS最適整列法に関してLOASとの性能比較を行い，それ等が実用的観点からLOASを凌ぐものでないことを示した[5]．ここで新たにLOASとの性能比較するのは下記2つの整列法である：

(1)MELソート：マージを基本とした整列法で，入力列をEncroaching List(以下Enc.List)と呼ばれる昇順列に分割し，マージする．Enc.Listの数を$Enc(\lambda)$とおけば$O(n \log Enc(\lambda))$で分割可能である．従って整列も$O(n \log Enc(\lambda))$となり，Enc最適である．

(2)SKIPソート：挿入整列法において，要素の挿入場所の探索に２分法を適用でき，かつ平均$O(n \log n)$で整列可能にするために，skip list[8]を用いたもの．これはたとえば$Runs$最適である．

これ等の新しい適応整列法に加えて，実用的なMERGEソート[7]とQUICKソート[6]も一緒に評価した．QUICKソートは軸選択をランダムに行うものとした．

３．評価方法と結果

各種整列法の性能評価をするために，2種類の方法で順列の列を生成した．1つの方法は理論的に一様に順列を生成するO(順列の長さ*葉数)のもの([4]の方式2)であり，他方は一様性の保証のない高速簡便法でO(数列の長さ)のもの([4]の方式3)である．両方式により葉数を1から2048まで変えながら長さ4095の順列を，葉数あたり50個ずつ生成した．そして各順列の整列に要するキーの比較回数と実行時間を計測した．そして葉数1の順列に対するLOASの計測結果によって他の葉数の計測結果を正規化した後図示した(図1,2,3)．各図における濃線が方式2[4]によって生成された順列に対する結果を示し，淡線は簡便法によるものを示す．

方式2の生成法は理論的には一様性が保証されているが，実際に大きなnに対する計算においては，桁落ちによる計算誤差が発生し，正しく順列が生成される保証はない．その場合に一様性を確認するためには，例えばn=4095のとき，約4000!のオーダーの自由度を持つ乱数列のランダムネスを調べる必要があり，我々はその方法を知らない．実際の評価結果(図1,2,3)を調べてみると，両方式による結果に大差はない．従って我々は次のような予想を立てている．「整列法の性能評価の目的で長い順列を生成するには，[4]の簡便法で十分である」．

図1は，必要なキーの比較回数を示している．

Comparison of Adaptive Sorting Algorithms,
Yoshihiko FUTAMURA, Koichi ENDO, Toshiharu HIRAI, Kenichi AOKI, School of Science & Engineering, Waseda University

キーの比較回数は,整列法の細かい実現法にあまり依存しないものである．従って大体の傾向として,SKIPソートやQUICKソートが無駄な比較を多く行っていることが解る(従来の性能比較では一様乱数列が用いられることが多く,その葉数は常に約$n/3$であり,QUICKに有利な偏った結論しか得ることができなかった)．図2は,キーの長さが小さいとき(2バイト程度)の実行時間の比較である．これによりQUICKソートやMELソートのオーバーヘッドの小ささが解る．図3は,キーの長さを50バイト程度に長くしたときの実行時間の比較である．キーの比較に時間が長くかかると,本質的に比較回数の少ないLOASが有利になる．特に葉数が少ない場合におけるLOASの高速性は顕著である．

4． おわりに

最新の適応整列法で,筆者等が有望視しているものと実用的な整列法の性能比較評価を,系統的に行った．図2,3,4から得られる結論は次の2点である:(1)キーサイズが小さいときは,葉数とほぼ無関係にQUICKソートが一番速い．(2)キーサイズが大きい時は葉数が大きくてもLOAS等の適応整列法が速い．整列法のより合理的な評価法を確立することが今後の課題である．

5． 参考文献

[1]浅野:各種ソーティングアルゴリズムの実際的評価,情報処理学会アルゴリズム研究会30-7,92年.

[2]COOK AND KIM:Best sorting algorithm for nearly sorted lists,Commun. ACM 23,1980,620-624.

[3]ESTIVILL-CASTRO AND WOOD:A survey of adaptive sorting algorithms,ACM Computing Surveys,Vol. 24,No. 4,December,1992,441-476.

[4]二村,青木,遠藤,大谷,白井:整列法評価のためのランダム順列の生成,情報処理学会第50回大会 6F-5, 1995.

[5]二村,二村,遠藤,平井:LOAS:葉数に関して最適な適応整列法,日本ソフトウェア科学会,94年11月.

[6]HOARE:Quicksort, Compt. J.,1,1,1962,10-15.

[7]KNUTH:The Art of Computer Programming Vol.3.

[8]PUGH:Skip lists:A probabilistic alternative to balanced trees, Commun. ACM,33,1990,668-676.

[9]SKIENA:Encroaching lists as a mesure of presortedness,BIT 28,1992,755-784.

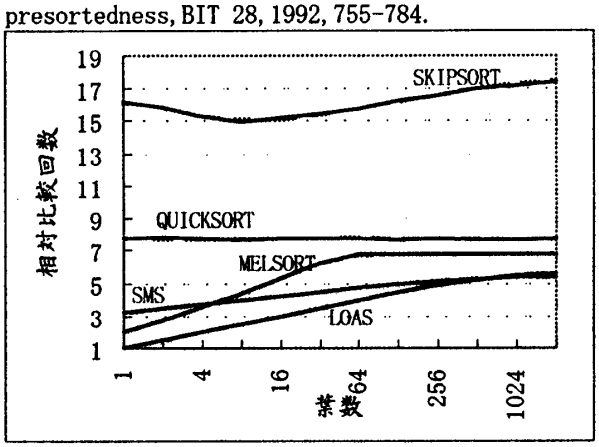

図1:長さ4095の順列の整列に要する比較回数
葉数1に対するLOASの比較回数(約8190)で正規化して示した．ただしSMSはMERGEソートを表す．

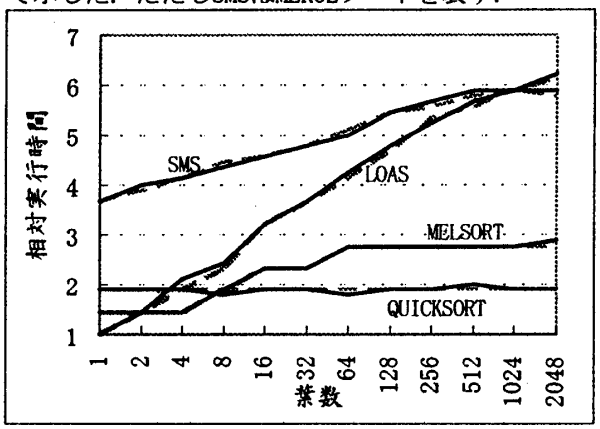

図2:長さ4095の2バイト長順列の整列時間
葉数1に対するLOASの実行時間で正規化して示した．

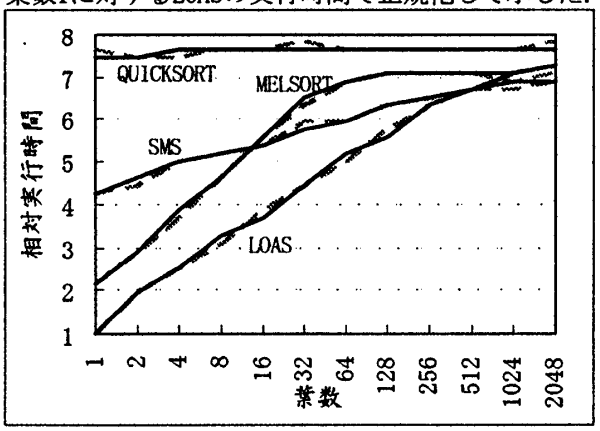

図3:長さ4095の50バイト長順列の整列時間
葉数1に対するLOASの実行時間で正規化して示した．