第3学年 AIスマート工学実験実習Ⅱ

プログラミングI 〜1~3週目〜

1. **二分探索木(Binary Search Tree)**
2. **説明**

親の左側のノードの値が必ず親よりも小さく、右側のノードの値が必ず親よりも大きくなっている二分木のことを二分探索木と呼ぶ。さらに、左の子だけでなく左部分木のすべてのノードの値が親よりも小さくなり、右部分木のすべてのノードの値も同様に親よりも大きくなる特性がある。

二分探索木では、それぞれのノードに対し、 ”左部分木のそれぞれのノード ＜ 基準ノード ＜ 右部分木のそれぞれのノード” が成り立つので、データの探索を簡単に行うことができる。

1. **プログラム**

以下に本実験で使用したプログラムと実行結果を示す。

テーブル が含まれている画像

自動的に生成された説明

テーブル

自動的に生成された説明

1. **考察**

親を基準に分割していくので、ある数を探すことなどには早いが親の値によっては実行時間にばらつきがある。その為、元々のデータによっては二分探索木の利点を活用できない場合がある。

1. **クイックソート(Quick sort)**
2. **説明**

　クイックソートというのは分割統治法というアルゴリズムの一種。まず、pivotという基準を決め、それを元にpivotより大きい値のグループと小さいグループに分け、また同じように基準をまた、グループ分けしていくのを行っていき最後全てのソートをくっつけることで並び替えることが出来る。

1. **プログラム**

以下に本実験で使用したプログラムと実行結果を示す。

テキスト

中程度の精度で自動的に生成された説明

1. **考察**

クイックソートは処理が早い、これは無駄な処理が少ないからであることが考えられる。2分割することで、小さい比較が出来るからであると思う。ただ、これもpivotが極端な値になると遅くなると思う。極端に慣ればなるほど分割してもあまり意味をなさなくなってしまうと思う。

1. **シェルソート(Shell sort)**
2. **説明**

挿入ソートは未整列の要素を一つずつ、整列済みの列の適切な位置に挿入していく手法。一定間隔で離れた要素をいくつかのグループに分けて、それぞれ整列させる。整列させたものをグループ分けした時の逆で1つの整列にする。これを繰り返すことで並び替えることが出来る。

1. **プログラム**

以下に本実験で使用したプログラムと実行結果を示す。

テキスト

中程度の精度で自動的に生成された説明

テーブル

自動的に生成された説明

1. **考察**

今まで使用してきたソートに関して(merge-sort, insertion-sort, bubble-sort, shell-sort, q-sort)実行時間を比較してソートに関して考察する。図1に5つのソートを使用した際の実行時間を示す。

グラフ, 折れ線グラフ

自動的に生成された説明グラフ, 折れ線グラフ

自動的に生成された説明

図1　5つのソートを使用した際の実行時間

図1から要素が10個の場合はbubble-sort, merge-sortが時間かかってしまうが、100個になるとmerge-sortより、insertion-sortの方が時間がかかっていることがわかる。図2にbubble-sort, merge-sortを使用した際の実行時間を示す。

グラフ, 折れ線グラフ

自動的に生成された説明グラフ, 折れ線グラフ

自動的に生成された説明

グラフ, 折れ線グラフ

自動的に生成された説明グラフ, 折れ線グラフ

自動的に生成された説明

図2　bubble-sortとinsertion-sort, merge-sortを使用した際の実行時間

　図2からbubble-sortと比べ、データ数が少ないときはinsertion-sortを使用し、多くのデータを取り扱うときはmerge-sortを使用した方が良いことが考えられる。図3にshell-sort, q-sortとinsertion-sort, merge-sortを

使用した際の実行時間を示す。

グラフ, 折れ線グラフ

自動的に生成された説明グラフ, 折れ線グラフ

自動的に生成された説明

グラフ, 折れ線グラフ

自動的に生成された説明グラフ, 折れ線グラフ

自動的に生成された説明

図3　shell-sort, q-sortとinsertion-sort, merge-sortを

使用した際の実行時間

図3からもデータ数が少ないときはinsertion-sortを使用し、多くのデータを取り扱うときはmerge-sortを使用した方が良いことが考えられる。他にもshell-sort, q-sort, merge-sortは実行時間が変わらず早いことが分かる。Insertion-sort, bubble-sortは数字を整列させる時に一つ一つ置き換えていることから他のソートに比べて時間がかかっていると思う。shell-sortのように分割することで半分以上もの時間を短縮できることが分かる。