خوارزميات الفرز و الترتيب

Selection Sort

Quick Sort

Radix Sort

تقديم

عبد الفتاح السباعي وأنس طارق ياسين

مقدمة

ليكن لدينا سلسلة تحتوي على n عنصر ، يرتبط بكل عنصر من عناصر السلسلة مفتاح ينتمي إلى مجموعة مرتبة كليا" .نريد الحصول على سلسلة تكون بديلا" لعناصر السلسلة الأصلية،بحيث تكون مفاتيح الفرز مرتبة تصاعديا" أو تنازليا" حين نقرأ السلسلة من البداية إلى النهاية. الغاية من الفرز هو الوصول السريع إلى المعلومات بطرق البحث .

يمكن اعتماد أي مكونات عناصر السلسلة مفتاح للفرز ، كما يمكن إيجاد تراكيب مختلفة من مكونات العناصر لإنشاء الفرز.

أي يمكن فرز مجموعة من الأشخاص بحسب الاسم أ، بحسب الكنية أ، بحسب العمر أو بحسب الطول. كما يمكن اعتماد ثنائيات مفتاحا" للفرز. سندرس مفاتيح الفرز هي عبارة عن قيم صحيحة ، و السلسلة التي نرغب بفرزها هي مجموعة من الأعداد الصحيحة أي العنصر يحتوي على المفتاح فقط و هذا لا يقلل من عمومية المسألة.

سوف ندرس خوارزميات فرز هي :

- ✓ الفرز بالاختيار Selection .
- ✓ الفرز Radix.
- ✓ الفرز السريع Quick.

خوارزمية الترتيب السريع Quick Sort

الترتيب السريع quicksort هي طريقة ترتيب من اختراع هوار (C.A.R.Hoare) في ١٩٦٢.

تعتمد الخوارزمية على وضع العنصر الأول (يسمى مؤشر) في مكانه النهائي ثم وضع العناصر الأكبر من المؤشر من جهة اليمين و العناصر الأصغر من جهة اليسار. و تسمى هذه العملية تجزئة. و بالنسبة لكل تحت-جدول، نحدد مؤشرا جديدا و نعيد عملية التجزئة. تتكرر هذه العملية إلى أن نحصل على مجموعة مرتبة.

إذا تم اختيار المؤشر بطريقة صحيحة، نحصل على الطريقة الأسرع للترتيب في الحالة المتوسطة، مع تعقيد بـ $O(n \log n)$ و التي قد تتحول إلى $O(n^2)$ في الحالة الأصعب، و هي حالة جدول عناصره مرتبة أصلا. و لكن هذه الحالة بديهية لأن المجموعة مرتبة أصلا.

في الناحية العملية، بالنسبة للتجزئات مع عدد قليل لا يتجاوز بضع عشرات من العناصر، يتم اللجوء عادة إلى الترتيب بالإدراج الذي يكون أفضل من الترتيب السريع. و بصفة عامة يعتبر الترتيب السريع الأكثر شيوعا (شعبية) من بين جميع خوارزميات الترتيب، و المشكلة الوحيدة تتمثل في كيفية اختيار المؤشر.

تعتبر من الخوارزميات الأسرع بشكل عام،و الأكثر تعقيدا" .

عند استعمال الترتيب السريع لترتيب مجموعة ذات عناصر كبيرة، يمكن تغيير تقنية الترتيب عند الوصول إلى مجموعة جزئية غير مرتبة عدد عناصرها صغير،١٠ أو أقل. الترتيب بالاختيار مناسب في هذه الحالة.

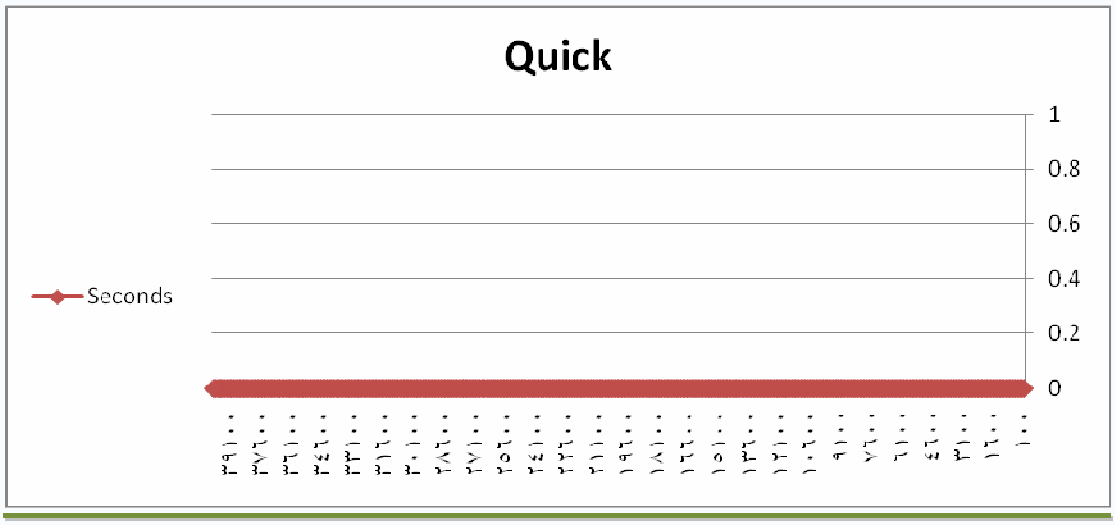
تتألف هذه الطريقة من أربع خطوات :

في حال وجود عنصر واحد فقط مراد ترتيبه ضمن اللائحة يتم إعادته مباشرة.

اختيار عنصر من اللائحة ليكون نقطة pivot للمصفوفة و عادة يكون أول عنصر من يسار اللائحة.

تقسيم اللائحة إلى نصفين أحدهما ذات قيم أصغر من القيمة المرجعية لنقطة pivot و النصف الآخر يحتوي على العناصر الأكبر من القيمة المرجعية لنقطة pivot.

تكرار العملية مع كل من النصفين اللذين تم تشكيلهم من المرحلة السابقة .



Selection Sort الترتيب بالاختيار

تقوم هذه الخوارزمية على بالبحث عن أكبر عنصر من السلسلة التي تحتوي على n عنصر التي عناصرها تبدأ من الدليل n-1 و حتى العنصر ذو الدليل indx ، و من ثم تضعه في المكان ذو الدليل n-1 من السلسلة المراد فرزها.

- يتم تقسيم السلسلة إلى قسمين ، القسم الأول عبارة عن سلسلة غير مفروزة Unsorted و القسم الثاني عبارة عن السلسلة المفروزة Sorted . في البداية القسم الغير مفروز هو عبارة عن السلسلة الأصلية و القسم الثاني فارغ.
- نأخذ دليل أول عنصر من السلسلة الغير مفروزة على أنه العنصر الأكبر.
- نقارن العنصر السابق مع بقية عناصر السلسلة غير المفروزة و نأخذ دليل العنصر الأكبر و من ثم نستبدله مع أول عنصر من السلسلة غير المفروزة ليصبح العنصر الأخير من السلسلة المفروزة و ينقص عدد عناصر السلسلة غير المفروزة بواحد.
- نستمر في الخطوتين السابقتين حتى يصبح عدد عناصر السلسلة الغير مفروزة واحد و الذي يمثل أصغر عنصر حيث نضعه أول عنصر من السلسلة المفروزة.

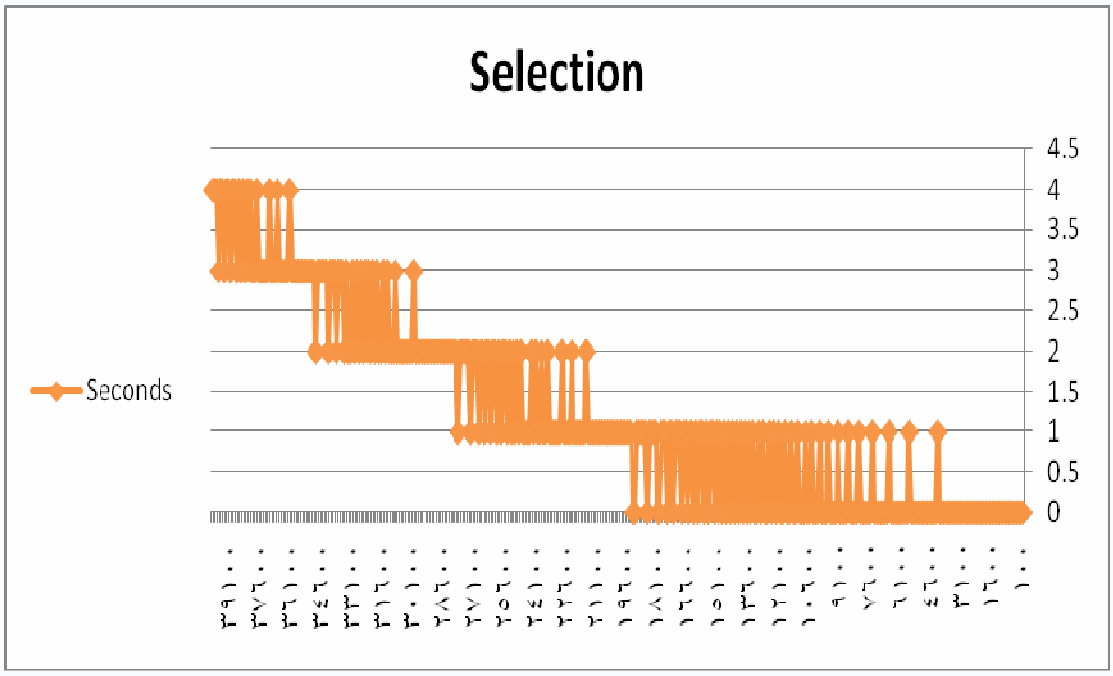
عدد المقارنات : من الملاحظ أن في كل عملية تكرارية لـ indx نقارن العنصر Array[indx] مع كافة العناصر Array[indx+1],.....Array[n-1] إذن لدينا n-indx مقارنة و بما أن عدد المقارنات لا يتعلق بمحتوى السلسلة فإن العدد الكلي للمقارنات يُعطى بالعلاقة التالية:

$$C=(n-1)+(n-)+....+2+1=(n(n-1)/2$$

عدد التبديلات : من الملاحظ أن عدد التبديلات لا يتعلق أيضا" بمحتوى السلسلة و العدد الكلي للتبديلات هو:

$$M_{min}=M_{max}=3(n-1)$$

إذن نستطيع القول أن تعقيد الخوارزمية بالاختيار هو $O(n^2)$.و هذه الخوارزمية لا تكون فعالة عندما يكون جزء من السلسلة مرتب.


Selection
Seconds
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0

الترتيب راديكس Radix Sort

Radix sort is a small method that many people intuitively use when alphabetizing a large list of names. (Here Radix is 26, 26 letters of alphabet). Specifically, the list of names is first sorted according to the first letter of each names, that is, the names are arranged in 26 classes. Intuitively, one might want to sort numbers on their most significant digit. But Radix sort do counter-intuitively by sorting on the least significant digits first. On the first pass entire numbers sort on the least significant digit and combine in a array. Then on the second pass, the entire numbers are sorted again on the second least-significant digits and combine in a array and so on.

Following example shows how Radix sort operates on seven 3-digits number.

INPUT	1st pass	2nd pass	3rd pass
329	720	720	329
457	355	329	355
657	436	436	436
839	457	839	457
436	657	355	657
720	329	457	720
355	839	657	839

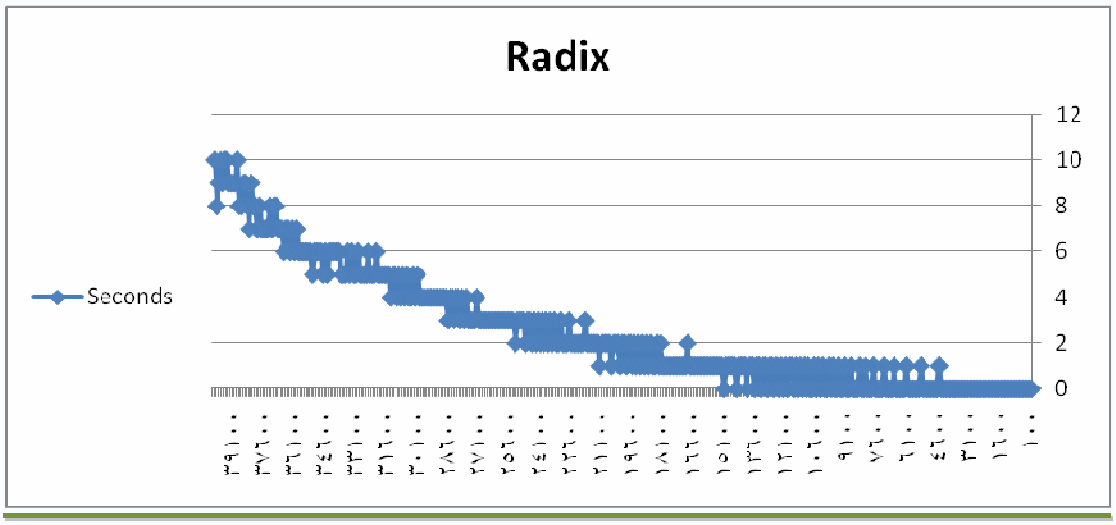
In the above example, the first column is the input. The remaining shows the list after successive sorts on increasingly significant digits position. The code for Radix sort assumes that each element in the n-element array A has d digits, where digit 1 is the lowest-order digit and d is the highest-order digit.

RADIX_SORT (A, d)

for $i \leftarrow 1$ to d do
    use a stable sort to sort A on digit i
    // counting sort will do the job

Analysis

The running time depends on the stable used as an intermediate sorting algorithm. When each digits is in the range 1 to k, and k is not too large, COUNTING_SORT is the obvious choice. In case of counting sort, each pass over n d-digit numbers takes $O(n + k)$ time. There are d passes, so the total time for Radix sort is $\Theta(n+k)$ time. There are d passes, so the total time for Radix sort is $\Theta(dn+kd)$. When d is constant and $k = \Theta(n)$, the Radix sort runs in linear time.



ملاحظة:

في المرفقات البرامج مكتوبة بلغة c++ وملف بالأعداد قبل الترتيب (طبعا الأعداد مولدة عشوائيا) وملف الأعداد بعد الترتيب . للمراسلة: goanose86@yahoo.com