Código Fuente

```
//Parámetros: Lista a ordenar, numero de dígitos del valor mayor de la lista y número de elementos de la lista
void radixSort(int list[], int digits, int size){
 //Ordena por dígitos, iniciando en 1 y terminando en el numero de dígitos"
    for (size_t actualDigit= 1; actualDigit<= digits; actualDigit++)
        sortByDigit(list, actualDigit, size);
}
void sortByDigit(int list[],int actualDigit, int size){
    int i, j, residue, aux;
    //Se inicializan los 10 "Buckets" en NULL o "vacío"
    struct node *zero = NULL, *one = NULL, *two = NULL, *three = NULL, *four = NULL;
    struct node *five = NULL, *six = NULL, *seven = NULL, *eight = NULL, *nine = NULL;
    //Se meten los 10 buckets a un arreglo de colas para un manejo más sencillo
    struct node *dec[10] = {zero, one, two, three, four, five, six, seven, eight, nine};
    //Desde 0 hasta el tamaño de la lista inicial
    //Cada valor de la lista, se agrega en el bucket
    //Que indique el digito en la posición de digito actual
    for (i = 0; i < size; i++){
        j = 1;
        if(actualDigit==1) residue = list[i] % 10;
        else residue = list[i];
        while(j<actualDigit) {
            residue /= 10;
            j++;
        }
        residue %=10; //Se obtiene el residuo
        //Si el bucket numero "residuo" del arreglo esta vacía, crea uno nuevo
        //si no, agrega el valor de lista[j] en la cola
        dec[residue] = (isEmpty(dec[residue])) ? createNode(list[i]) : queue(dec[residue], list[i]);
    }
    aux = 0;    //Se reinicia el valor auxiliar
    //este indica el índice de la lista a manipular
    for (j = 0; j < size; j++){
        while (!(isEmpty(dec[j])))
          //Se agrega a la lista en el índice indicado por aux y aumenta aux
           dec[j] = dequeue(dec[j], &list[aux++]);
    }
}
```

Correo Electrónico

Gomez.Emmanuel@uabc.edu.mx

Página web

https://sites.google.com/uabc.edu.mx/radixsort

Referencias

Ordenamiento Radix. Es.wikipedia.org. (2020). Retrieved 13 December 2020, from https://es.wikipedia.org/wiki/Ordenamiento_Radix.

profile, V. (2020). Radix Sort. Algoritmo-radix-sort.blogspot.com. Retrieved 13 December 2020, from http://algoritmo-radix-sort.blogspot.com/p/home.html.

Radix sort. En.wikipedia.org. (2020). Retrieved 13 December 2020, from https://en.wikipedia.org/wiki/Radix_sort.

Algoritmos y Estructura de Datos

Método de ordenamiento Radix

Alumno: Gómez Cárdenas Emmanuel Alberto

Matricula: 1261509

Docente: Palacios Guerrero Alma Leticia

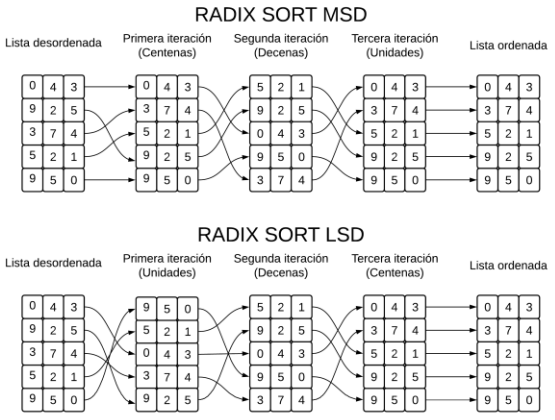
Fecha: 14 de diciembre de 2020

Radix Sort

RadixSort es un algoritmo para el ordenamiento de datos con llaves enteras, agrupando las llaves por sus dígitos individuales que comparten la misma posición y valor.

Gracias a que los enteros pueden ser usados para representar cadenas, este método también funciona para otros tipos de datos, además de los enteros.

RadixSort no está basado en hacer comparaciones entre los elementos por lo tanto no está limitado y puede llegar a ejecutarse en un tiempo linear (depende de la implementación).



Orden de los dígitos

Existen dos clasificaciones del método de ordenamiento Radix: MSD y LSD.

MSD Radix: MSD está dado por las siglas en ingles de digito más significativo, por lo tanto, este método empieza a ordenar la lista desde el digito más significativo hasta llegar al digito menos significativo (LSD). Este método es el más utilizado cuando se quieren ordenar cadenas o enteros de una longitud fija. Por ejemplo, una cadena [r, f, as, rs, y, a] seria ordenada como [a, as, f, r, rs, y] y si se ordenara una cadena de enteros con longitud variable del 0 al 11 y se usa un orden lexicográfico estos terminarían ordenados [0, 1, 10, 11, 2, 3, 4, 5, 6, 7, 8, 9] como si las llaves cortas fueran justificadas a la izquierda y llenadas con espacios para hacer las llaves cortas del mismo tamaño que las largas.

LSD Radix: Esta clasificación utiliza el orden de ordenamiento siguiente: las llaves cortas van antes que las largas, y las llaves del mismo tamaño son ordenadas lexicográficamente el cual coincide con el orden de la representación de enteros. Como lo es la secuencia [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11].

Origen

Surgió gracias a una idea de Herman Hollerith (1860-1929), creador de la máquina tabuladora, la cual ordenaba tarjetas perforadas dependiendo de la ubicación de las últimas 3 columnas. Esta máquina fue utilizada para el censo de la población de Estados Unidos en 1880.

Tiempo de ejecución.

Debido a que el algoritmo no hace comparaciones, no tiene nada de importancia el estado inicial de la lista a ordenar. Por lo tanto, el mejor y peor caso cuentan con el mismo tiempo de ejecución, que siempre depende del número de dígitos del valor más grande. Suponiendo que tenemos “n” números y que cada número puede tener “w” posibles valores, Radix sort ordena en O(nw).