Федеральное агентство связи

Сибирский Государственный Университет Телекоммуникаций и

Информатики

СибГУТИ

Кафедра ПМиК

Лабораторная работа №12.1

«Слияние серий»

Выполнил: студент 1 курса группы ИП-013

Иванов.Л.Д

Проверил: старший преподаватель кафедры ПМиК

Солодов П.С.

Новосибирск

2021

Лабораторная работа № 12-2

1). Разработать процедуру сортировки списка методом прямого слияния (MergeSort)..Для проверки правильности сортировки использовать сервисные процедуры для работы со списками, разработанные в лабораторной работе № 11.

2). Предусмотреть подсчет фактического количества

пересылок и сравнений (Мф и Сф), сравнить с теоретическими оценками М и С.

2). Проверить работу метода прямого слияния на массивах убывающих, возрастающих и случайных чисел и сделать вывод о зависимости или независимости метода от исходной упорядоченности массива. Построить таблицу вида:

**код:**

#include <stdlib.h>

#include <stdio.h>

#include <ctime>

struct spis

{

int Data;

spis \*next;

} \*a, \*b, \*c, \*ro, \*S;

void VoidQueue(spis \*tail, int n)

{

for(int i=0;i<n;i++)

{

ro=new spis;

ro->Data=NULL;

tail->next=ro;

tail=ro;

}

tail->next=NULL;

}

void IncQueue(spis \*tail, int n)

{

for(int i=0;i<n;i++)

{

ro=new spis;

ro->Data=i;

tail->next=ro;

tail=ro;

}

tail->next=NULL;

}

int KolSpis(spis \*head)

{

int sum=1;

for(ro=head->next; ro; ro=ro->next)

if(ro->next)

sum++;

return sum;

}

void PrintSpis(spis \*head)

{

for(ro=head->next; ro; ro=ro->next)

printf("%d ",ro->Data);

printf("\n");

}

int Merge(int q, int r, spis \*a, spis \*b, spis \*c)

{

a=a->next;

b=b->next;

c=c->next;

while((q)&&(r))

{

if(a->Data <= b->Data)

{

c->Data=a->Data;

c=c->next;

a=a->next;

q--;

}

else

{

c->Data=b->Data;

c=c->next;

b=b->next;

r--;

}

}

while(q)

{

c->Data=a->Data;

q--;

if(q)

{

c=c->next;

a=a->next;

}

}

while(r)

{

c->Data=b->Data;

r--;

if(r)

{

c=c->next;

b=b->next;

}

}

c->next=NULL;

}

void Crush(spis \*S, spis \*a, spis \*b, int n)

{

spis \*k,\*p;

a=S; b=S->next; n=1;

k=a; p=b;

while(p)

{

n++;

k->next=p->next;

k=p;

p=p->next;

}

}

//void MergeMerge(spis \*S, int n)

//{

// spis \*C[1];

// int r,i,m,ro,q,pp;

// Merge(S,a,b,n);

// r=1;

// while(r<n)

// {

// VoidQueue(C[0],n);

// VoidQueue(C[1],n);

// i=0; m=n;

// while(m>0)

// {

// if(m>=pp) q=r; else q=m;

// m=m-q;

// if(m>=pp) ro=r; else ro=m;

// m=m-ro;

// MergeSort(q,r,a,b,C[i]);

// i=1-i;

// }

// a=C[0].Head; b=C[1].Head;

// pp=2\*pp;

// }

// C[0].Tail->next=NULL;

// S=C[0].Head;

//}

int main()

{

int q,r;

S=new spis;

S->next=NULL;

IncQueue(S,10);

a=new spis;

a->next=NULL;

b=new spis;

b->next=NULL;

c=new spis;

c->next=NULL;

PrintSpis(S);

Crush(S,a,b,10);

//

// VoidQueue(c,10);

//

// q=KolSpis(a);

// r=KolSpis(b);

//

// Merge(q, r, a, b, c);

// printf("\n");

// int f[n],Is[n],Sh[n];

//

// for(int i=0; i<5; i++)

// {

//

// n = (i+1)\*100;

// M=C=0;

// FillInc(A,n);

// f[i]=PiramSort(A, n);

// M=C=0;

// FillDec(A,n);

// Is[i]=PiramSort(A, n);

// M=C=0;

// FillRand(A,n);

// Sh[i]=PiramSort(A, n);

// }

//

// printf(" \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

// printf("| n | MergeSort (M+C) |\n");

// printf("|\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\n");

// printf("| | Inc | Dec | rand |\n");

// printf("|\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_|\n");

// for(int i=0; i<5; i++)

// {

// printf("| %d | %d | %d | %d \n",(i+1)\*100,f[i],Is[i],Sh[i]);

// printf("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\n");

// }

//

return 0;

}

Вывод: Сегодня я изучил алгоритм сортировки Merge Sort.