Чепасов В.И., Харченко Д.А., Шкут А.С.
Оренбургский государственный университет

ПОЛУЧЕНИЕ СВЯЗАННЫХ ПЕРЕСТАНОВОК В ЗАДАЧЕ О РАСПИСАНИИ

Один из способов оптимизации расписания заключается в следующем: для каждого варианта следования дисциплин получают соответствующий вариант расписания и из всех вариантов расписания выбирают тот, в котором число распределенных часов будет максимальным. Для реализации такой оптимизации необходимо получить множество связанных перестановок для определения множества вариантов следования дисциплин.

Связанные перестановки – это совокупность перестановок, в которой каждая последующая перестановка получается из предыдущей одной перестановкой (переставляются два элемента).

Рассмотрим идею алгоритма получения связанных перестановок.

Возьмем три элемента: 1, 2, 3.

Соответственно три сечения: 1, 2, 3.

Вводим два массива номеров для сечений 2 и 3:

$S2_i$ – массив номеров для второго сечения,

$S3_i$ – массив номеров для третьего сечения.

Берем первому элементу массива $S2_1$ присваиваем 2 (элемент второго сечения в исходной перестановке): $S2_1 = 2$.

Сравниваем первый элемент массива $S2_1$ с элементом в первом сечении 1.

Они не равны.

Поэтому меняем их местами: 2, 1, 3.

Второму элементу $S2_2$ присваиваем 1:

$S2_2 = 1$.

Сравниваем элемент в первом сечении 2 с элементами массива второго сечения $S2_1 = 1$, $S2_2 = 2$.

Поскольку элемент 2 в первом сечении равен второму элементу массива $S2_2$, то переходим к третьему сечению.

Элементу $S3_1 = 3$.

Переставляем элемент из третьего сечения 3 с элементом 2 в первом сечении (можно с элементом 1 во втором сечении): 3, 1, 2.

Определяем $S3_2 = 2$.

Осуществляем перестановку первого и второго элементов по уже рассмотренной схеме: 1, 3, 2.

Сравниваем элементы в 1-м и 2-м сечениях с элементами массива $S3_1 = 3$, $S3_2 = 2$.

Поскольку 1 не равен ни одному элементу массива S3, то меняем местами 2 и 1: 2, 3, 1.

При этом $S3_3 = 1$.

Меняем местами первый и второй элементы по уже рассмотренной схеме: 3, 2, 1.

Все перестановки для трех элементов получены:

1, 2, 3
2, 1, 3
3, 1, 2
1, 3, 2
2, 3, 1
3, 2, 1

Если бы был четвертый элемент (четвертое сечение), то аналогично вводим массив $S4_i$, в который заносим по указанной схеме элементы четвертого сечения, которые будут помещаться в это сечение в результате перестановок.

Например, в первом рассмотрении $S4_1 = 4$, во втором рассмотрении $S4_2$ может быть любым из трех элементов 1, 2, 3.

Например, $S4_2 = 1$, потом $S4_3 = 2$, потом $S4_4 = 3$.

Так для перестановки:

3, 2, 1, 4.

Начинаем осуществлять перестановки по указанной схеме для 2-го и третьего сечений.

То есть получим шесть перестановок.

Далее в четвертое сечение помещаем любой из трех элементов предыдущих сечений 1, 2, 3. Например, 1.

Начинаем перестановки со второго, потом с третьего сечений.

Получаем шесть перестановок.

Потом в 4-е сечение помещаем 2 и т. д.

После заполнения вектора четвертого сечения $S4_i$ переходим по указанной схеме к пятому сечению и так для всех сечений.

Очевидно, все перестановки будут разными и связанными.

В представленной блок-схеме:

ia_i – элементы перестановки,

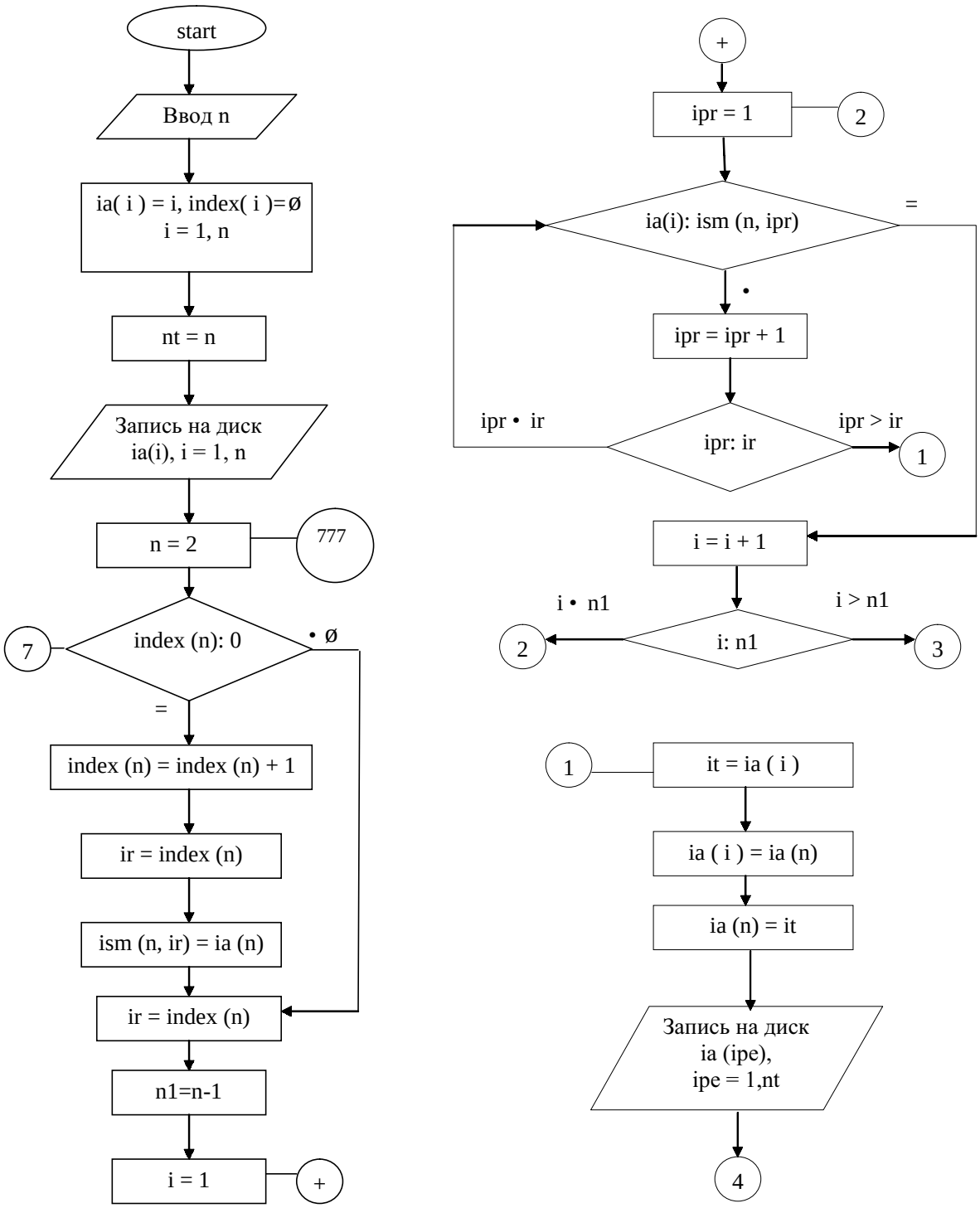
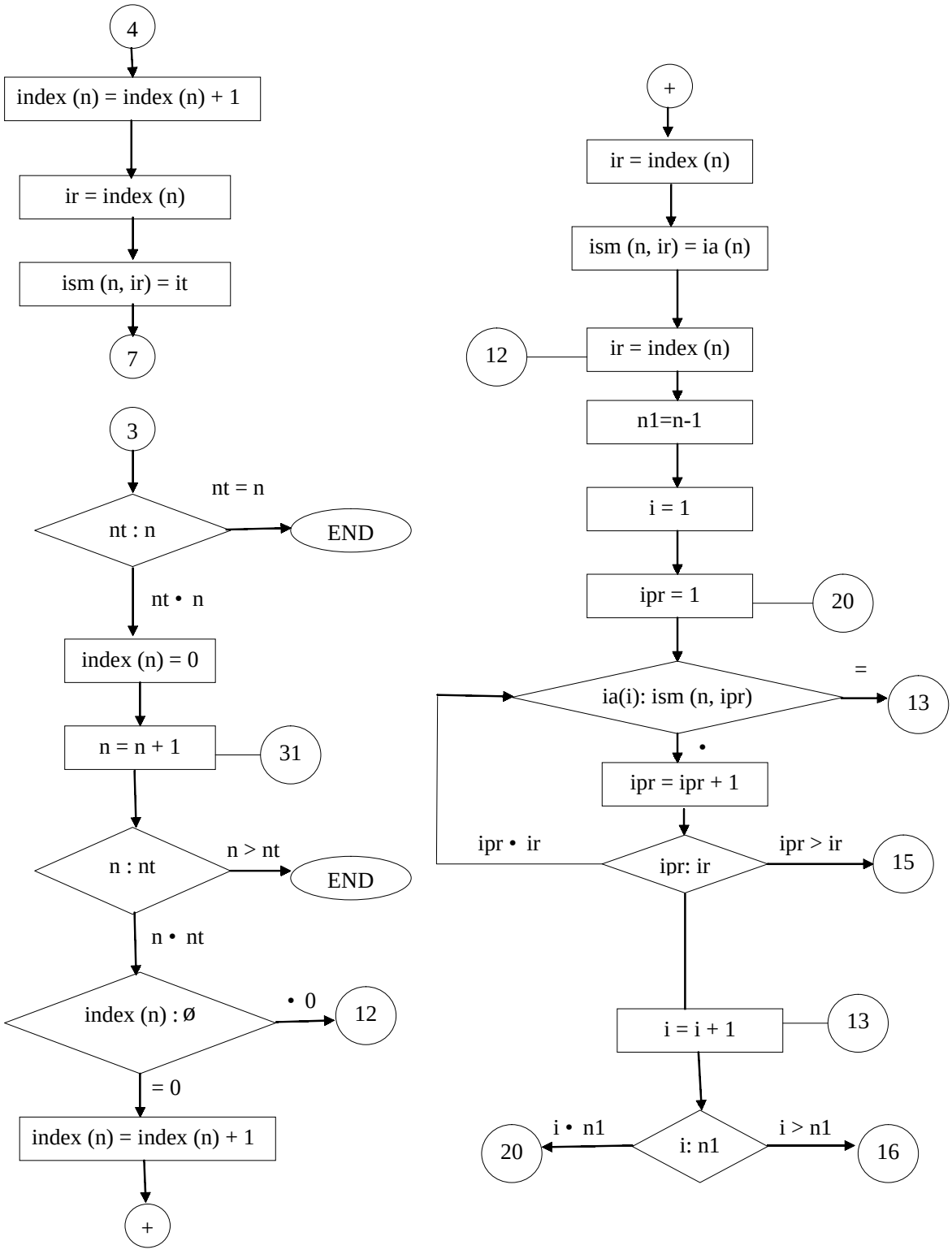
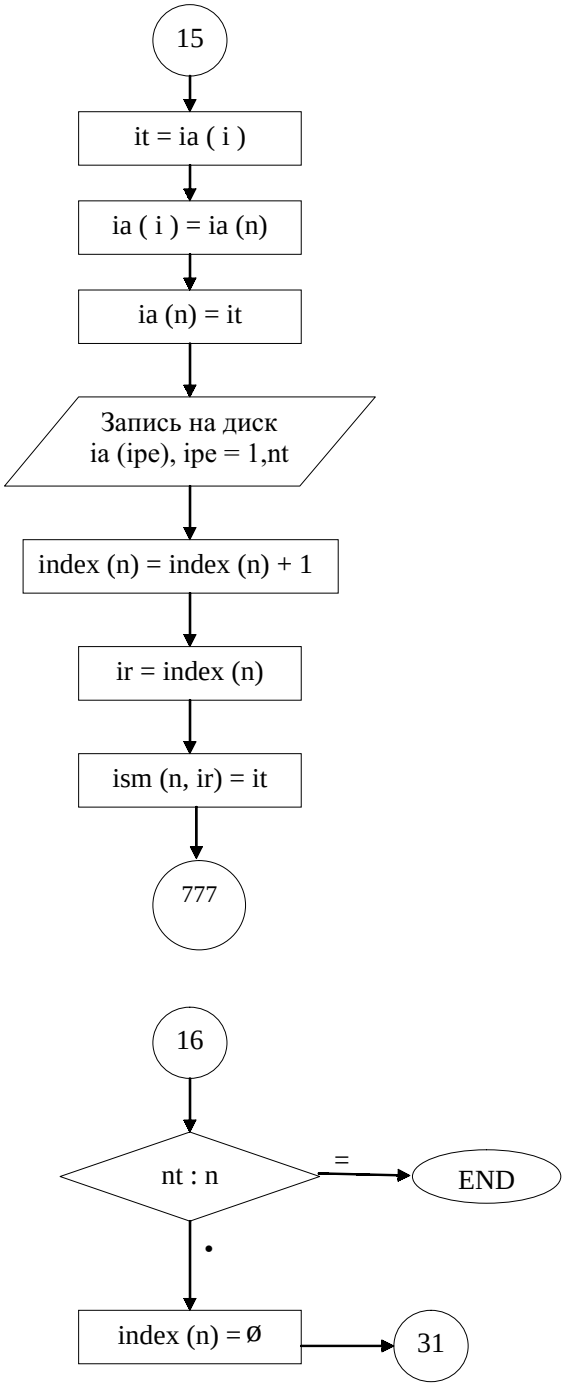


Рисунок. Блок-схема представленного алгоритма



Продолжение рисунка. Блок-схема представленного алгоритма



Продолжение рисунка. Блок-схема представленного алгоритма

$index_n$ – счетчик элементов в n-м сечении,
$ism_{n, ipr}$ – массив номеров элементов в n-м сечении,
n – номер сечения,
ipr – индекс для просмотра номеров элементов в n-м сечении.

Текст программы по получению и проверке неповторяемости связанных перестановок:

```
#include<stdio.h>
main()
{
FILE *fp5,*fp6,*fp7;

     int ia[31],is2[31],is3[31],
     ib[31],ipe,n1,iprow,il,ip,is,
      j,jp,it1,ich,i1,
     index[31],ism[31][31],n,nt,i,ir,ipr,
     itot,it,kper;
     printf(«input n<=30\n»);
     scanf(«%d»,&n);
     printf(«введите количество
     перестановок для записи\n»);
     scanf(«%d»,&kper);
     nt=n;
     for(i=1;i<=n;i++)
     {
     ia[i]=i;
     index[i]=0;
     }
     itot=0;
     itot=itot+1;
     fp6=fopen(«swjzanc»,»w»);
     fp7=fopen(«perstanc»,»w»);
     for(ipe=1;ipe<=nt;ipe++)
     {
     fprintf(fp6,»%d «,ia[ipe]);
     fprintf(fp7,»%d «,ia[ipe]);
     }
     fprintf(fp6,»\n»);
     fprintf(fp7,»\n»);
m777: n=2;
m7: if(index[n]!=0) goto m2;
     index[n]=index[n]+1;
     ir=index[n];
     ism[n][ir]=ia[n];
     m2:;
     ir=index[n];
     n1=n-1;
     for(i=1;i<=n1;i++)
     {
     for(ipr=1;ipr<=ir;ipr++)
     if(ia[i]==ism[n][ipr]) goto m3;
```

```
    goto m5;
    m3:;
    }
    goto m6;
    m5:;
    it=ia[i];
    ia[i]=ia[n];
    ia[n]=it;
    itot=itot+1;
    for(ipe=1;ipe<=nt;ipe++)
    {
    fprintf(fp6,»%d «,ia[ipe]);
    fprintf(fp7,»%d «,ia[ipe]);
    }
    fprintf(fp6,»\n»);
    fprintf(fp7,»\n»);
    if(itot==kper) goto m1999;
    index[n]=index[n]+1;
    ir=index[n];
    ism[n][ir]=it;
    goto m7;
m6:;
    if(nt==n) goto m1999;
    index[n]=0;
m5678:;
    n=n+1;
    if(n>nt) goto m1999;
    if(index[n]!=0) goto m12;
    index[n]=index[n]+1;
    ir=index[n];
    ism[n][ir]=ia[n];
m12:;
    ir=index[n];
    n1=n-1;
    for(i=1;i<=n1;i++)
    {
    for(ipr=1;ipr<=ir;ipr++)
    if(ia[i]==ism[n][ipr])
    goto m13;
    goto m15;
m13:;
    }
    goto m16;
m15:;
    it=ia[i];
    ia[i]=ia[n];
    ia[n]=it;
    itot=itot+1;
    for(ipe=1;ipe<=nt;ipe++)
    {
    fprintf(fp6,»%d «,ia[ipe]);
    fprintf(fp7,»%d «,ia[ipe]);
    }
    fprintf(fp6,»\n»);
    fprintf(fp7,»\n»);
    if(itot==kper) goto m1999;
    index[n]=index[n]+1;
    ir=index[n];
    ism[n][ir]=it;
    goto m777;
m16:;
    if(nt==n) goto m1999;
    index[n]=0;
    goto m5678;

m1999:;
    fprintf(fp6,»количество
    записанных перестановок=%d\n»,
    itot);
    fclose(fp6);
    fclose(fp7);
    printf(«если делаем проверку на
    частоту встречаемости,\n»);
    printf(«то вводим 1,если
    нет,то вводим 0\n»);
    scanf(«%d»,&iprow);
    if(iprow==0) goto m2999;
    fp6=fopen(«prowerc»,»w»);
    fp5=fopen(«swjzanc»,»r»);
    il=itot;
    for(i=1;i<=il;i++)
    {
    for(ip=1;ip<=nt;ip++)
    fscanf(fp5,»%d»,&ia[ip]);
    fp7=fopen(«perstanc»,»r»);
    is=0;
    for(j=1;j<=il;j++)
    {
    for(ip=1;ip<=nt;ip++)
    fscanf(fp7,»%d»,&ib[ip]);
    for(jp=1;jp<=nt;jp++)
    if(ia[jp]!=ib[jp]) goto m235;
    is=is+1;
m235:;
    }
    fclose(fp7);
    if(is==1) goto m234;
    fprintf(fp6,» перестановка-%d
    частота=%d\n»,i,is);
    for(jp=1;jp<=nt;jp++)
    fprintf(fp6,»%d «,ia[jp]);
    fprintf(fp6,»\n»);
m234:;
    }
    fclose(fp6);
    fclose(fp5);
m2999:;
    fp5=fopen(«swjzanc»,»r»);
    fp6=fopen(«cepcpp»,»w»);
    it1=itot-1;
```

```
    ich=0;
    for(i=1;i<=it1;i++)
    {
    if(ich!=0) goto m3007;
    for(ip=1;ip<=nt;ip++)
    fscanf(fp5,»%d»,&ia[ip]);
    for(ip=1;ip<=nt;ip++)
    fscanf(fp5,»%d»,&ib[ip]);
    for(it=1;it<=nt;it++)
    is3[it]=ib[it];
    ich=1;
    goto m3008;
m3007:;
    for(it=1;it<=nt;it++)
    ia[it]=is3[it];
    for(ip=1;ip<=nt;ip++)
    fscanf(fp5,»%d»,&ib[ip]);
    for(it=1;it<=nt;it++)
    is3[it]=ib[it];
m3008:;
    is=0;
    for(ip=1;ip<=nt;ip++)
    {
    if(ia[ip]==ib[ip]) goto m3002;
    is=is+1;
    is2[is]=ip;
m3002:;
    }
    if(is==2) goto m3003;
    fprintf(fp6,» отличий больше
    двух\n»);
m3003:;
    for(ip=1;ip<=is;ip++)
    fprintf(fp6,»%d «,is2[ip]);
    fprintf(fp6,»\n»);
m3001:;
    }
    i1=0;
    fprintf(fp6,»%d%d\n»,i1,i1);
    fclose(fp5);
    fclose(fp6);
    }
```

Из созданного файла связанных перестановок можно создать файл номеров переставляемых элементов при обходе связанных перестановок.

Этот файл номеров переставляемых элементов будет использоваться для оптимизации расписания путем перестановок в списке дисциплин.

Список использованной литературы:
1. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. М.: МЦНМО, 2000. - 960 с.
2. Шень А. Программирование: теоремы и задачи. М.: МЦНМО, 1995. - 264 с.