ROMAI J., **1**, 2(2005),
149–155

INTERPOLATION POLYNOMIALS. APPLICATION IN FINDING SOME COMBINATORIAL FORMULAS

Doru Popescu Anastasiu
Univesity of Piteşti, Faculty of Mathematics and Computer Sciences
dopopan@yahoo.com

Abstract A way of solving some counting problems using interpolation polynomials, starting with some values determined by the backtracking method is found. The obtained formulas are implicitly present in the programmes that solve the problems.

1. THE INTERPOLATION WITH FINITE DIFFERENCES

We intend to solve the following

Problem 1.1. *Given m points in the plane, $A_i(x_i, y_i)$, i=1,2,...,m, find the polynomial function associated with a polynomial of degree m-1, the graph of which contains the given points.*

Theorem 1.1. *Let $A_i(x_i, y_i)$, i=1,2,...,m be m points in the plan, such that $x_i \neq x_j$ for every $i \neq j$. Then there exists a unique polynomial P of maximum degree m-1 so that the graph of the polynomial function associated with the polynomial contains the given points.*

The proof of the theorem can be found in [4].

The polynomial P, whose existence and uniqueness follow from the theorem, is called the interpolation polynomial through the points $A_i(x_i,y_i)$, i=1,2,...,m.

From now on $g(X; x_1,...,x_m)$ will denote the interpolation polynomial through the points $A_i(x_i, y_i)$, i=1,2,...,m.

Also, $C(x_1, x_2, \ldots, x_m)$ denotes the coefficient of X^{m-1} from $g(X; x_1, x_2, \ldots, x_m)$. The number $C(x_1, x_2, \ldots, x_m)$ is called the *finite difference* associated with the points $A_1, A_2, \ldots, A_m$. We divide the polynomial $g(X; x_1, x_2, \ldots, x_m)$ by $(X-x_1)(X-x_2)\ldots(X-x_{m-1})$ and using the equality $g(X; x_1, x_2, \ldots, x_m) = C(x_1, x_2, \ldots, x_m).\ (X-x_1)(X-x_2)\ldots(X-x_{m-1}) + R(X)$, where R(X) is a polynomial of maximum degree m-2, we obtain

$R(X) = g(X; x_1, x_2, \ldots, x_m) - C(x_1, x_2, \ldots, x_m).\ (X-x_1)(X-x_2)\ldots(X-x_{m-1})$. So $R(x_i)=y_i$, i=1,2,...,m, thus R(X) is an interpolation polynomial through the points $A_1, A_2, \ldots, A_{m-1}$. In this way we obtained the following formula

$g(X; x_1, x_2, \ldots, x_m) = g(X; x_1, x_2, \ldots, x_{m-1}) + C(x_1, x_2, \ldots, x_m).\ (X-x_1)(X-x_2)\ldots(X-x_{m-1})$,
which will be applied recursively to
$g(X; x_1, x_2, \ldots, x_{m-1})$, $g(X; x_1, x_2, \ldots, x_{m-2})$, $\ldots$, $g(X; x_1)$.

The last interpolation polynomial $g(X;x_1)$ is actually the constant polynomial $g(X;x_1)=g_1$. Thus: $g(X; x_1, x_2, \ldots, x_m)=C(x_1)+C(x_1, x_2)\cdot(X-x_1)+\ldots+ C(x_1, x_2, \ldots, x_m).\ (X-x_1)(X-x_2)\ldots(X-x_{m-1})$, which is called the *Newton's formula of interpolation with finite differences.*

Let us compute the finite differences $C(x_1)$, $C(x_1, x_2),\ldots$, $C(x_1, x_2, \ldots, x_m)$. To this aim let us consider the polynomial

$$f(X) = \frac{X - x_m}{x_1 - x_m} \cdot g(X; x_1, ..., x_{m-1}) + \frac{x_1 - X}{x_1 - x_m} \cdot g(X; x_2, x_3, ..., x_m)$$

It has the following properties

$$1)\ f(x_1) = \frac{x_1 - x_m}{x_1 - x_m} \cdot g(x_1; x_1, ..., x_{m-1}) + \frac{x_1 - x_1}{x_1 - x_m} \cdot g(x_1; x_2, x_3, ..., x_m) = y_1$$

$$2)\ f(x_m) = y_m$$

$f(x_i) = \frac{x_i - x_m}{x_1 - x_m} \cdot g(x_i; x_1, ..., x_{m-1}) + \frac{x_1 - x_i}{x_1 - x_m} \cdot g(x_i; x_2, x_3, ..., x_m) = y_i$,for i=2,...,m-1.

Remark that f is an interpolation polynomial through the points (x_1,y_1), (x_2, y_2), $\ldots$, (x_m, y_m). Due to the fact that the interpolation polynomial is unique, it follows that

$$g(X; x_1, ..., x_m) = \frac{X - x_m}{x_1 - x_m} \cdot g(X; x_1, ..., x_{m-1}) + \frac{x_1 - X}{x_1 - x_m} \cdot g(X; x_2, x_3, ..., x_m).$$

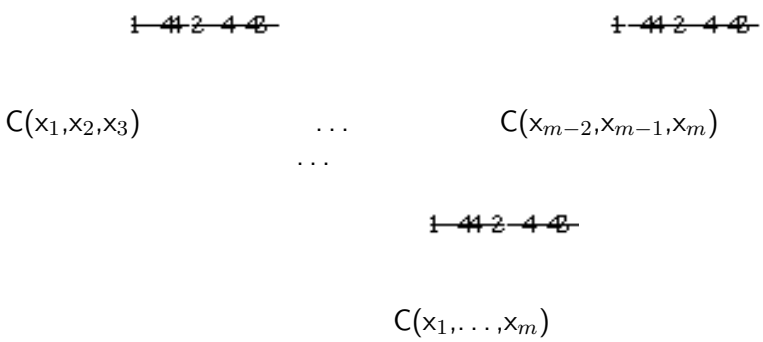
By identifying the coefficients of the highest degree, we obtain

$$C(x_1, ..., x_m) = \frac{C(x_1, ..., x_{m-1}) - C(x_2, ..., x_m)}{x_1 - x_m}.$$

This formula shows that the finite differences can be computed using the following diagram

$C(x_1)=y_1$ $C(x_2)=y_2$ $C(x_3)=y_3$ $\ldots$ $C(x_{m-1})=y_{m-1}$ $C(x_m)=y_m$

$C(x_1,x_2)$ $C(x_2,x_3)$ $\ldots$ $C(x_{m-1},x_m)$



Remark 1.1. *Taking a closer look to the diagram, we notice that every column with finite differences is determined using the column prior to the current one. This fact will be used in the program which determines the interpolation polynomial with finite differences.*

2. SOLVING THE COUNTING PROBLEMS

In this section we solve the following type of counting problems

Given n a positive integer, different from zero, and, eventually, other input data, determine the number of vectors x=($x_1, x_2, \ldots, x_n$), which satisfy some conditions.

Remark 2.1. *If the number wanted is a polynomial formula which depends on n, then the problem can be solved using the following algorithm:*

We determine k values of the number wanted $y_1, y_2, \ldots, y_k$, using the backtracking method (for n=1, n=2, . . . , n=k).
sw←0; j←1
repeat
We determine the interpolation polynomial P for the points (1, y_1), (2, y_2),. . . , (j, y_j)
sw←1
for *i←j+1, k* **do**
if *P(i)¡¿y_i* **then**
sw←0
break
endif
endfor
j←j+1
until *(sw=1)***or***(j¿k)*
if *j¡=k* **then**
read *n*
write *P(n)*

else
write *"Is necesary more number y with backtracking method"*
endif

The complexity of the algorithm

-in the first part of the algorithm (y_1, y_2, ..., y_k), the complexity depends on the input data, having an exponential growth because of the backtracking method;
-the second part of the algorithm has a complexity of $O(k^3)$, because the algorithm that finds the interpolation polynomial with finite differences for k points is executed in $O(k^2)$.

The correctness of the algorithm

If we consider that the formula of counting the solutions is of the following type: $f(n)=a_p n^p + a_{p-1}n^{p-1} + \ldots + a_1 n + a_0$, p¡k and P(n) is an interpolation polynomial which follows after exiting the while cycle, then we obtain
$f(1)=y_1$
$f(2)=y_2$
...
$f(k)=y_k$
$P(1)=y_1$
$P(2)=y_2$
...
$P(k)=y_k$

Form these equalities it follows that the polynomial f-P has as roots the numbers 1, 2, ..., k. From the fact that the polynomials f and P have degrees smaller than k, it follows that the polynomial f-P is null. Thus, f=P. In this case, any value computed by the interpolation polynomial gives the exact solution (if the calculations do not exceed the predefined types; otherwise, operations with big numbers must be implemented).

3. APPLICATION

Given n, a positive integer from the interval [1,1000], how many positive integers of n digits have the product of their digits equal to 8 ?

Exemple. For n=2 we obtain 4 numbers.

Solution. Let x_1, x_2, ..., x_n be the digits number (x_1 is the first digit). From $x_1x_2\ldots x_n=8$ and x_1, x_2, ..., x_2 are digits we have three events:
$x_i=x_j=x_k=2$ and $x_h=1$, $h\notin\{i,j,k\}$;
$x_i=2$, $x_j=4$, and $x_h=1$, $h\notin\{i,j\}$;
$x_i=8$, $x_h=1$, $h\neq i$.

We obtain the wanted number : $\begin{pmatrix} n \\ 3 \end{pmatrix} + A_n^2 + n = \frac{1}{6}n^3 + \frac{1}{2}n^2 + \frac{1}{3}n$ (*).
Having a value for n, we can compute the wanted number using the formula (*).

Another way of solving the problem, without explicitly finding the formula, proceeds as follows. With the backtracking method, we determine the wanted number for the first values of n (for exmple for n=1,...,5) and then, using the interpolation polynomial with finite differences, we solve the problem. The following C++ program determines the value of the interpolation polynomial with finite differences for the given n, and compares the found value with the one from the formula (*).

```
#include <iostream.h>
#include <conio.h>
int sw,j,i,m,cif[4]={1,2,4,8},v[10];
float aux,x0,c[1000],p;
long x[10],y[10],numara,n;
long produs(int k){
long i,p;
p=1;
for(i=1;i<=k;i++)
p*=v[i];
return p;
}
void back(int k){
int i;
long p;
if (k==n+1)
{
if (produs(n)==8)
numara++;
}
else
for(i=0;i<4;i++)
{
v[k]=cif[i];
p=produs(k);
if ((p!=0)&&(p<=8))
back(k+1);
}
}
void puncte(int m){
int i;
for(i=1;i<=m;i++)
{
n=i;
numara=0;
x[i]=i;
back(1);
y[i]=numara;
}
}
float polinom_cu_d_f(float x0, int m){
```

```
int i,k,p,pas;
float c1[100],c2[100],f,temp;
for(i=1;i<=m;i++)
c1[i]=y[i];
p=m;pas=1;
c[1]=y[1];
for(k=2;k<=m;k++)
{
for(i=1;i<p;i++)
c2[i]=(c1[i]-c1[i+1])/(x[i]-x[i+pas]);
c[k]=c2[1];
for(i=1;i<=p;i++)
c1[i]=c2[i];
pas++;
p–;
}
f=0;
temp=1;
for(i=1;i<=m;i++)
{
f+=c[i]*temp;
temp*=(x0-x[i]);
}
return f;
}
void main(){
puncte(5);
for(i=1;i<=5;i++)
cout<<x[i]<<" "<<y[i]<<"\n";
j=1;
do{
sw=1;
for(i=j+1;j<=5;j++){
p=polinom_cu_d_f(i,j);
if (p!=y[i])
{
sw=0;
break;
}
}
j++;
} while((sw==0)&&(j<5));
if (sw)
{
cout<<"n=";
cin>>n;
p=polinom_cu_d_f(n,j-1);
cout<<"With interpolation polynomials with f. d. ";
cout<<p<<"\n";
aux=n*(n+1)*(n+2)/6;
```

```
cout<<"With formula "<<aux<<"\n";
}
getch();
}
```

4. TASKS

1. What is the maximum number of sides, in which the plane can be divided using:

n lines ?

n circles ?

where n is a positive integer smaller than 1000.

2. How many rectangles (different in shape or position), consisting of an integer number of squares, can be drawn on a nxn chess board? (n<300).

3. The same task as 2, if we replace rectangles with squares.

4. Given a nxn chess board (n<10000, n even), find the maximum number of kings that can be placed on the board, without attacking themselves.

5. Given a nxn chess board (n<10000, n multiple of 3), find the maximum number of knights that can be placed on the board, without attacking themselves.

6. Given two positive integers different from zero, n and k, find how many binary sequences (a_1,...,a_n) have exactly $k$monotone sequences.

References

[1] A.M. Iaglom, I.M. Iaglom, Probleme neelementare tratate elementar. Ed. Tehnică, Bucureşti, 1983.

[2] Matematica pe mapamond, Bacău, 1995.

[3] D. E. Knuth, Arta programării calculatoarelor, vol. 1,3, Ed. Bucureşti, 2002.

[4] G.D. Mateescu, I.C. Mateescu, Analiză numerică, Ed. Petrion, Bucureşti, 1995.

[5] H. S. Wilf, Algorithms and Complexity, Internet Edition, 1994.

[6] I. Parberry, Lecture notes on algorithm analysis and computational complexity, Internet Edition, 2001.

[7] T. Cormen, C. Leiserson, R. Rivest, Introducere în algoritmi, Ed. Libris-Agora, Cluj-Napoca, 2000.

[8] B.P. Demidovich, I.A. Maron, Computational mathematics, MIR, Moscow, 1987.