PROBLEM 2: SEPARATING FUNCTIONS

TEAM: FRANCE 3

.

Abstract

We expose the simplest methods we have found to solve questions 1 to 3b, which heavily rely on Bezout's theorem. For the generalization of question 3a, we wondered if the fact that $n-1$ numbers among the $a_1, \ldots, a_n$ are divisible by p implies that $F(a_1, \ldots, a_n)$ is divisible by p. In question 5, we give means to calculate $F(a_1, \ldots, a_n)$ is some particular cases. Relying on numerical simulations, we propose two conjectures. More precisely, one of the conjectures gives a closed formula for $F(a_1, a_3, a_3)$ when the integers a_1, a_2, a_3 are consecutive terms of an arithmetic progression.

1 Exercice 2

1.1 Question 1

Let $(a_1, \ldots, a_n)$ be n positive integers such that $\gcd(a_1, a_2, \ldots, an) = 1$ and set

$$S = \{a_1x_1 + \cdots a_nx_n, x_i \in \mathbb{N}\}.$$

By Bezout's theorem, there exist n integers $u_1, \ldots, u_n$ such that $a_1u_1 + \cdots + a_nu_n = 1$. Let:

$$A^+ = \sum_{\{i; u_i > 0\}} a_iu_i$$
$$A^- = -\sum_{\{i; u_i < 0\}} a_iu_i.$$

We now show that any integer n such that $n \geqslant A^-(A^- + 1)$ belongs to S. Remark that A^+ and A^- belong to S and moreover $A^+ - A^- = 1$. Using the Euclidean division by A^- , we see that every positive integer n can be written as follows:

$$\begin{aligned} n &= kA^- + r, \qquad \text{with } k \geqslant 0,\ A^- > r \geqslant 0 \\ &= kA^- + r(A^+ - A^-) \\ &= (k - r)A^- + rA^+ \\ &= (k - r)A^- + r(A^- + 1). \end{aligned}$$

Thus if $n \geqslant A^-(A^- + 1)$, then necessarily $k - r \geqslant 0$ and $n \in S$.

1.2 Question 2

Let a_1 and a_2 two positive coprime integers. We show that $F(a_1, a_2) = (a_1 - 1)(a_2 - 1)$. Let $N = F(a_1, a_2) - 1$ (which is well defined by question 1.), so that $N + a_1 \in S$ and thus there exist non negative integers x_1, x_2 such that

$$N + a_1 = a_1x_1 + a_2x_2.$$

Hence $N = a_1(x_1 - 1) + a_2x_2$. But $N \notin S$, implying $x_1 - 1 < 0$. This means $x_1 < 1$, and since x_1 is nonnegative, this implies $x_1 = 0$. It follows that N can be written in the following form:

$$N = -a_1 + a_2x,$$

where $x = x_2 \geq 0$. A similar argument shows that N can be written $N = a_1y - a_2$ with $y \geq 0$. Consequently:

$$-a_1 + a_2x = a_1y - a_2,$$

i.e.

$$a_2(x+1) = a_1(y+1).$$

By Gauss's lemma, since a_1 and a_2 are coprime, there exists an integer k such that $y+1 = ka_2$, so that:

$$\begin{aligned} N &= a_1(a_2k-1) - a_2 \\ &= a_1(a_2(k-1)-1) + a_2(a_1-1). \end{aligned}$$

However $N \notin S$, which forces $k=1$, implying $N = a_1a_2 - a_1 - a_2$. As a consequence, $F(a_1,a_2) = N+1 = (a_1-1)(a_2-1)$.

1.3 Question 3

1.3.1 Question 3a

The integers a_1, a_2, a_3 are globally coprime. Since a_1 and a_2 are even, a_3 is odd. Let $N = F(a_1,a_2,a_3) - 1$. Then there exist nonnegative integers x_1, x_2, x_3 such that

$$N + a_3 = a_1x_1 + a_2x_2 + a_3x_3.$$

Proceeding as in question 2, we deduce that:

$$N = a_1x_1 + a_2x_2 - a_3.$$

Hence N is odd and $F(a_1,a_2,a_3) = N+1$ is even.

1.3.2 Question 3b

Recall that by hypothesis (c.f the statement) $d = \gcd(a_1,a_2)$ and $a_3 \geq F(\frac{a_1}{d}; \frac{a_2}{d})$.

- Consider the case $d=1$, that is a_1, a_2 are coprime. We know that there exist nonnegative integers (x_1,x_2,x_3) such that $F(a_1,a_2,a_3) = a_1x_1 + a_2x_2 + a_3x_3$. On the one hand $F(a_1,a_2,a_3) \leqslant F(a_1,a_2)$ and on the other hand if $x_3 > 0$, $a_1x_1 + a_2x_2 + a_3x_3 \geqslant a_3 \geqslant F(a_1,a_2)$. Thus $F(a_1,a_2,a_3) = F(a_1,a_2) = dF(a_1,a_2) + F(d,a_3)$. If $x_3 = 0$ then $F(a_1,a_2,a_3) = a_1x_1 + a_2x_2$ and we are back to question 2 (1.2).
- Now suppose $d > 1$. Set $a_1' = \frac{a_1}{d}$, $a_2' = \frac{a_2}{d}$ so that $\gcd(a_1', a_2') = 1$. Let $G = dF(a_1',a_2') + F(d,a_3)$. We first show that for any $n \in \mathbb{N}$, $G+n$ belongs to S. Let y, x_3 be nonnegative integers such that $F(d,a_3) + n = dy + a_3x_3$ and let x_1, x_2 be nonnegative integers such that $F(a_1',a_2') + y = x_1a_1' + x_2a_2'$. Then:

$$\begin{aligned} G+n &= dF(a_1',a_2') + F(d,a_3) + n \\ &= dF(a_1',a_2') + dy + x_3a_3 \\ &= d(F(a_1',a_2') + y) + x_3a_3 \\ &= d(x_1a_1' + x_2a_2') + x_3a_3. \end{aligned}$$

This shows that $G+n \in S$ and consequently the inequality $F(a_1, a_2, a_3) \leqslant G$.

It now remains to show that $G - 1 \notin S$. We reason by contradiction and suppose that $G - 1$ can be written as following:

$$G - 1 = d(x_1 a_1' + x_2 a_2') + x_3 a_3,$$

with $x_1, x_2, x_3 \geq 0$. But by question 2, $F(d, a_3) = (d - 1)(a_3 - 1)$, which gives:

$$G - 1 = dF(a_1', a_2') + a_3 d - a_3 - d,$$

so that $x_3 a_3 \equiv -a_3 \bmod d$, in other words $a_3(x_3+1) \equiv 0 \bmod d$. Since a_3 and d are coprime, $x_3 = kd - 1$ for some integer $k \geq 1$. Equating and simplifying by d we obtain, using $F(a_1', a_2') \leq a_3$:

$$\begin{aligned} a_1' x_1 + a_2' x_2 + k a_3 &= F(a_1', a_2') + a_3 - 1 \\ &< 2a_3 \end{aligned}$$

It follows that $k = 1$. Hence $F(a_1', a_2') - 1 = a_1' x_1 + a_2' x_2$, which is impossible by definition of $F(a_1', a_2')$. The desired contradiction has been obtained, giving $G - 1 \notin S$ and finally $G = F(a_1, a_2, a_3)$.

1.3.3 Remark

Note that the preceding property is not a sufficient condition. For instance, for $(a_1, a_2, a_3) = (7, 11, 25)$ we have $d = \gcd(a_1, a_2) = 1$. Even if $60 = F(7, 11, 25) = dF(a_1/d, a_2/d) + F(d, a_3)$, the inequality $a_3 \geq F(a_1/d, a_2/d)$ is false.

1.4 Question 4

We are interested in the following question. Fix a prime number p. If $n - 1$ numbers among the globally coprime integers $a_1, \ldots, a_n$ are divisible by p, is $F(a_1, \ldots, a_n)$ necessarily divisible by p?

1.4.1 Case $p = 2$

The idea of 3a) remains valid: we claim that if $a_1, a_2, \cdots, a_n$ are globally coprime and $a_1, \cdots a_{n-1}$ are even, then $F(a_1, a_2, \cdots, a_n)$ is even.

We turn to the proof. As before, remark that a_n is odd. Let $N = F(a_1, a_2, \cdots, a_n) - 1$. Then

$$N + a_n \in S \quad \textit{and} \quad N + a_n = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n,$$

for some $x_i \geqslant 0$. As in question 3a we deduce that:

$$N = a_1 x_1 + a_2 x_2 + \cdots - a_n.$$

Thus N is odd and $F(a_1, a_2, \cdots, a_n) = N + 1$ is even.

1.4.2 General p

Suppose that $a_1, \ldots, a_{n-1}$ are divisible by p. Use once again the previous formula:

$$F(a_1, \cdots, a_n) - 1 = a_1x_1 + a_2x_2 + \cdots + a_{n-1}x_{n-1} - a_n.$$

Then modulo p this turns into:

$$F(a_1, \cdots, a_n) - 1 = -a_n \bmod p.$$

Hence $F(a_1, \cdots, a_n)$ is divisible by p if, and only if $a_n = 1 \bmod p$.

1.5 Question5

We give means to calculate $F(a_1, a_2, \cdots, a_n)$ in some particular cases.

1.5.1 Generalization of question 3b

Similarly to question 3b), we see that for n globally coprime positive integers $(a_1, \ldots, a_n)$, if $d = gcd(a_1, a_2, \cdots, a_{n-1})$, then d and a_n are coprime and if $a_n \geqslant F(a'_1, a'_2, \cdots, a'_{n-1})$ (where for $1 \leq i \leq n-1$, a'_i stands for a_i/d) then:

$$F(a_1, a_2, \cdots, a_n) = dF(a'_1, a'_2, \cdots, a'_{n-1}) + F(d, a_n).$$

1.5.2 The case of arithmetic progressions

In this section, we give a conjecture to calculate $F(a_1, \ldots, a_n)$ when the numbers a_i belong to some arithmetic progression.

We are driven by the following two motivations. On the one hand, n integers being fixed, we can always find an arithmetic progression containing these numbers. On the other hand, prime numbers can be written under the form $N = 6n + 1$ or $N = 6n - 1$.

From numerical simulations, it stems that (except for 3-uplets for which $F(a_1, a_2, a_3)$ is given by the formula of question 3b) for three consecutive terms of an arithmetic progression:

$$A = 1 + nr \qquad B = 1 + r + nr \qquad C = 1 + 2r + nr,$$

where $n, r \geq 0$ are integers, we have

$$F(A, B, C) = \frac{r^2}{2}n^2 + \frac{2r^2 - r}{2}n.$$

Similar formulas exist for three consecutive terms belonging to an arithmetic progression which does not contain 1. We tried to find closed formulas in the case where the three integers belong to an arithmetic progression but are not consecutive terms, and in the case $n = 4$ for consecutive terms of an arithmetic progression, but we did not succeed.

1.5.3 Case when two couples are never coprime

Fix an integer $n \geq 3$ and let $p_1 < \cdots < p_{2n-3}$ be distinct prime numbers. We draw our attention on n-tuples of the following form:

$$a_1 = p_1 p_2 \cdots p_{n-1}, \qquad \cdots \qquad a_n = p_n p_{n+1} \cdots p_{2n-3} p_1.$$

For instance, for $n = 3$ these triplets are:

$$a_1 = p_1 p_2, \quad a_2 = p_2 p_3, \quad a_3 = p_3 p_1$$

for three distinct prime numbers $p_1 < p_2 < p_3$.

We conjecture the following formula:

$$F(a_1, \ldots, a_n) = 1 + \sum_{1 \leq i_1 < i_2 < \cdots < i_n \leq 2n-3} p_{i_1} p_{i_2} \cdots p_{i_n} - \sum_{i=1}^{n} a_i.$$

For $n = 3$, this can be proved by using question 3.b, but we did not manage to find a proof for other values of n.

For example, for n=4 and

$$a_1 = p_1 p_2 p_3, a_2 = p_2 p_3 p_4, a_3 = p_3 p_4 p_5, a_4 = p_4 p_5 p_1,$$

we have:

$$F(a_1, a_2, a_3, a_4) = p_1 p_2 p_3 p_4 + p_2 p_3 p_4 p_5 + p_3 p_4 p_5 p_1 - a_1 - a_2 - a_3 - a_4 + 1.$$

We have obtained the following by numerical simulations, which are consistent with the preceding formula.

- $F(3 * 5 * 7, 5 * 7 * 11, 7 * 11 * 13, 11 * 13 * 3) = 7244$,
- $F(29 * 31 * 37, 31 * 37 * 41, 37 * 41 * 43, 41 * 43 * 29) = 5080996$ (one hour of work for the computer !).

1.6 Script of the program

C.f. the annexe.

1.7 How did we make the conjectures?

We used a spreadsheet like Excel. For example, we obtained the following results:

- For even n and $A = 1 + 3n \;\; B = 4 + 3n \;\; C = 7 + 3n \rightarrow \;\; F(A, B, C) = \frac{9}{2}n^2 + \frac{15}{2}n$,

- for even n and $A = 1+7n \;\; B = 8+7n \;\; C = 15+7n \rightarrow \;\; F(A,B,C) = \frac{49}{2}n^2 + \frac{91}{2}n$,
- for even n and $A = 1+9n \;\; B = 10+9n \;\; C = 19+9n \rightarrow \;\; F(A,B,C) = \frac{81}{2}n^2 + \frac{153}{2}n$,
- for odd n and $A = 1+6n \;\; B = 7+6n \;\; C = 13+6n \rightarrow \;\; F(A,B,C) = 18n^2 + 33n$,
- For all n and $A = 2+3n \;\; B = 5+3n \;\; C = 8+3n \rightarrow \;\; F(A,B,C) = \frac{9}{2}n^2 + \frac{21}{2}n + 3$,
- For all n and $A = 2+5n \;\; B = 7+5n \;\; C = 12+5n \rightarrow \;\; F(A,B,C) = \frac{25}{2}n^2 + \frac{55}{2}n + 5$,
- For all n and $A = 3+5n \;\; B = 8+5n \;\; C = 13+5n \rightarrow \;\; F(A,B,C) = \frac{25}{2}n^2 + \frac{65}{2}n + 11$.

This led us to the conjecture exposed in section 1.5.2.

PROGRAM FOR N=3

```
#include <stdio.h>
#include <math.h>





int pgcd (int a, int b)                    // fonction pgcd = algo d'euclide
        {
        int r = 0;

        while (b != 0)
                {
                r = a % b;
                a = b;
                b = r;
                }
        return a;
        }


int minimum3 (int min1, int min2, int min3) // on definit la fonction minimum de trois nombres
        {
        int F = 0;

        if ( min1 <= min2 && min1 <= min3)
                        {
                        F = min1;
                        }

        if ( min2 <= min1 && min2 <= min3)
                        {
                        F = min2;
                        }

        if ( min3 <= min1 && min3 <= min2)
                        {
                        F = min3;
                        }

        return F;
        }




int main (int argc, const char * argv[])

{
// déclaration des variables
int A = 0;
int B = 0;
int C = 0;
int i = 0;

printf ("\nentrez les valeurs dans l'ordre croissant : "); // ordre croissant pour simplifier et accélérer

printf ("\n valeur de A : ");
```

```c
scanf ("%d", &A);

printf ("\n valeur de B (supérieure à A) : ");
scanf ("%d", &B);

printf ("\n valeur de C (supérieure à B): ");
scanf ("%d", &C);

printf ("\n\nVous avez choisi : \nA : %d \nB : %d \nC : %d", A, B, C);

int X = pgcd (A, B);
int Y = pgcd (A, C);
int Z = pgcd (B, C);

printf ("\npgcd (A, B) : %d \npgcd (A, C) : %d \npgcd (B, C) : %d \n\n", X, Y, Z);

int DebutDeLaBoucle = 0;
int MinimumTrouve = 0; // 0 ==> le minimum n'est pas trouvé / 1 ==> minimum trouvé = fin de la boucle

int F = 0;

// F(a,b,c) = d(f(a/d ; b/d)) + F(c,d) et les circulaires

int min1 = X * ((A / X)-1) * ((B / X)-1) + (X-1) * (C-1);
int min2 = Y * ((A / Y)-1) * ((C / Y)-1) + (Y-1) * (B-1);
int min3 = Z * ((B / Z)-1) * ((C / Z)-1) + (Z-1) * (A-1);




printf ("\nmin1 : %d", min1);
printf ("\nmin2 : %d", min2);
printf ("\nmin3 : %d", min3);

F = minimum3 (min1, min2, min3);

printf ("\nF : %d\n", F);

int K = 0;
int R = 0;
int negatif = 0; // 0 ==> positif / 1 ==> négatif
int k = 0;

while ( MinimumTrouve == 0) //début grande boucle
        {


        i=0;
        printf ("\ndebut de la boucle \n");

        F = F - 1;
        DebutDeLaBoucle = 0;

        printf ("F = %d", F);


        // étape 1 : on vérifie si le nombre ne s'écrit pas comme combinaison linéaire de A et B.

        // on vérifie si ce n'est pas un multiple du pgcd

        while ( DebutDeLaBoucle == 0)

        {
```

```
            if ( (F - i*C) < 0 )
            {
                DebutDeLaBoucle = 1 ;
                        MinimumTrouve = 1 ;
            }

            if ( (F - i*C) % X == 0 && DebutDeLaBoucle == 0)
                                {

                                printf ("\n entrée dans l'étape 1\n");

                                K = 0;
                                negatif = 0;

                                while ( K < A ) // boucle while 1
                                            {

                                            R = (F - i*C - (K*B)) % A;
                                            printf ("F - i*C - (K*B):%d\n",F - i*C - (K*B));
                                            // on verifie que c'est positif

                                            if ( (F - i*C - (K*B)) < 0 )
                                                        {
                                                                    printf ("\n entrée dans négatif 1\n");
                                                        K = A; // sortie de la petite boucle while 1
                                                        negatif = 1; // pour ne pas rentrer dans le reste de la boucle
                                                        }

                                            if ( R == 0 && negatif == 0)
                                                        {
                                                        printf ("\npas d'entrée dans négatif 1\n");
                                                        DebutDeLaBoucle = 1; // ==> sortie de la moyenne boucle
                                                        K = A; // sortie de la petite boucle while 1
                                                        }

                                            K++ ;
                                            printf ("K++\n");
                                            }
                                i++;
             printf ("i++\n");
                                }

            }}


printf ("minimum : %d", F+1);
scanf ("%d",&F);
return 0;
}
```

PROGRAM FOR N=4

```
#include <stdio.h>
#include <math.h>


int pgcd (int a, int b)                    // fonction pgcd = algo d'euclide
        {
        int r = 0;

        while (b != 0)
                {
                r = a % b;
                a = b;
                b = r;
                }
        return a;
        }

int minimum3 (int min1, int min2, int min3) // on definit la fonction minimum de trois nombres
        {
        int F = 0;

        if ( min1 <= min2 && min1 <= min3)
                        {
                        F = min1;
                        }

        if ( min2 <= min1 && min2 <= min3)
                        {
                        F = min2;
                        }

        if ( min3 <= min1 && min3 <= min2)
                        {
                        F = min3;
                        }

        return F;
        }

int FcombAB (int F, int A, int B)
{
        int d = 0;
        int i;
        int G = 0;
        if ( (F % pgcd(A,B)) != 0 )
        {
                return 0;
        }
        else
        {
                for ( i=0; d==0 ; i++)
                {
                        G=F-i*A;
                  if (G < 0 )
                                {
                                        return 0;
                                        d=1;
                                }
                        else if (G % B == 0)
                                {
                                        return 1;
                                        d=1;
```

```
                                        }
                }
            }
}

int FcombABC (int F, int A, int B, int C)
            {
            int d = 0;
            int j = 0;
            int G = 0;

            while ( d==0 )
                        {
                                    G = F-j*C;
                                    if (G < 0 )
                                                {
                                                            return 0;
                                                            d=1;
                                                }
                                    else if(FcombAB(G,A,B)== 1)
                                                {
                                                            return 1;
                                                            d=1;
                                                }
                j++;
                }
}

int FcombABCD (int F, int A, int B, int C, int D)
{
            int d = 0;
            int i;
            int G = 0;

            for ( i=0; d==0 ; i++)
                        {

                  G=F-i*D;
                                    if (G < 0 )
                                                {
                                                            return 0;
                                                            d=1;
                                                }
                                    else if(FcombABC(G,A,B,C)== 1)
                                                {
                                                            return 1;
                                                            d=1;
                                                }

                }
}

int main (int argc, const char * argv[])

{

int A = 0;
int B = 0;
int C = 0;
int D = 0;
int i = 0;
int min1 = 0;
int min2 = 0;
```

```c
int min3 = 0;
int X=0,Y=0,F=0,Z=0;

printf ("\nentrez les valeurs dans l'ordre croissant : "); // ordre croissant pour simplifier et accélérer
printf ("\n valeur de A : ");
scanf ("%d", &A);
printf ("\n valeur de B (supérieure à A) : ");
scanf ("%d", &B);
printf ("\n valeur de C (supérieure à B) : ");
scanf ("%d", &C);
printf ("\n valeur de D (supérieure à C) : ");
scanf ("%d", &D);

X=pgcd(A,B);
Y=pgcd(A,C);
Z=pgcd(B,C);
 min1 = X * ((A / X)-1) * ((B / X)-1) + (X-1) * (C-1);
 min2 = Y * ((A / Y)-1) * ((C / Y)-1) + (Y-1) * (B-1);
 min3 = Z * ((B / Z)-1) * ((C / Z)-1) + (Z-1) * (A-1);
F = minimum3 (min1, min2, min3);

while (FcombABCD(F,A,B,C,D)== 1)
{
F--;
printf ("F = %d\n",F);
}
printf ("F = %d",F);
scanf ("%d",&F);
}
```