Assignment 1

Maxime CHAMBREUIL
maxime.chambreuil@mail.mcgill.ca

Contents

1 Exercise 1

$$\begin{cases} x \equiv 13 \pmod{35} \\ x \equiv 11 \pmod{36} \\ x \equiv 23 \pmod{37} \end{cases}$$

To use the Chinese Remainder Theorem [1], we have to check 35, 36 and 37 are primes 2 by 2, by decomposing them and look for a common factor :

[1] cf Bibliography

$$35 = 5 \times 7$$
$$36 = 2^2 \times 3^2$$
$$37 = 37$$

So we can use the Chinese Remainder Theorem, which tells us that x_0 is the unique solution of our problem :

$$x_0 = u_1(36 \times 37)13 + u_2(35 \times 37)11 + u_3(35 \times 36)23 + 35 \times 36 \times 37n$$

$n, u_1, u_2, u_3 \in Z$
Now, our job now consists in finding u_i.

1.1 Finding u_1

As 35, 36 and 37 has no common factor, we can say that it exists 2 numbers u_1 and $u_1^{'}$ which enable us to write :

$$36 \times 37 \times u_1^{'} + 35 \times u_1 = 1$$

$$36 \times 37 = 1332 = 35 \times 38 + 2$$
$$35 = 2 \times 17 + 1$$

So

$$\begin{aligned} 1 &= 35 - 2 \times 17 \\ 1 &= 35 - (1332 - 35 \times 38)17 \\ 1 &= 35(1 + 38 \times 17) - 17 \times 1332 \\ 1 &= 35 \times 647 - 17 \times 1332 \end{aligned}$$

We can conclude that

$$u_1 = -17$$

1.2 Finding u_2 and u_3

By repeating the same process but with adapting the numbers we find that $u_2 = -1$:

$$1 = 36 \times 36 - (35 \times 37)$$

and $u_3 = -18$:

$$1 = 37 \times (1 + 34 \times 18) - 18(35 \times 36)$$

Finally,

$$\begin{aligned} x &= -17(36 \times 37)13 - (35 \times 37)11 - 18(35 \times 36)23 + (35 \times 36 \times 37)n \\ x &= -294372 - 14245 - 521640 + 46620n \\ x &= 46620n - 830257 \\ x &= 46620(n - 17) - 37717 \\ x &= 46620(n - 18) + 8903 \end{aligned}$$

In conclusion, the lowest solution to our problem is :

$$\boxed{\text{x = 8 903}}$$

2 Exercise 2 : From Brassard-Bratley's book

2.1 Proof about rootLV algorithm

The property we have to prove is :

$$\text{rootLV finds } \sqrt{x} \iff \text{rootLV chooses a, that gives the key to } \sqrt{x}.$$

I assume that you have the rootLV algorithm with you. Let us begin with the left hand side :

$$\begin{aligned}
\text{rootLV finds } \sqrt{x} &\iff a^2 \equiv x \bmod p \text{ or } c = 0 \\
&\iff p | a^2 - x \text{ or } (a+\sqrt{x})^{\frac{p-1}{2}} \equiv d\sqrt{x} \bmod p \\
&\iff (\frac{a^2-x}{p}) = 0 \text{ or } (a+\sqrt{x})^{\frac{p-1}{2}} \equiv d\sqrt{x} \bmod p
\end{aligned}$$

Thanks to the hand out from the book, we know that if $a+\sqrt{x}$ is put to the power of $\frac{p-1}{2}$, then its legender symbol is 1 or - 1. The hand out tell us also that if $(a+b)$ belongs to QR_p (respectively QNR_p), then $(a-b)$ belongs to QNR_p (respectively QR_p). We can generalize that idea by saying that $(\frac{a+\sqrt{x}}{p})(\frac{a-\sqrt{x}}{p}) = -1$. So our proof goes on :

$$\begin{aligned}
\text{rootLV finds } \sqrt{x} &\iff (\frac{a^2-x}{p}) = 0 \text{ or } (\frac{a+\sqrt{x}}{p})(\frac{a-\sqrt{x}}{p}) = -1 \\
&\iff (\frac{a^2-x}{p}) = 0 \text{ or } (\frac{a^2-x}{p}) = -1 \\
&\iff (a^2-x) \bmod p \notin QR_p \\
\text{rootLV finds } \sqrt{x} &\iff \text{rootLV chooses a, that gives the key to } \sqrt{x}
\end{aligned}$$

2.2 Number of choices

3 Exercise 3

4 Exercise 4 : Jacobi Symbol Algorithm

4.1 Notations

$$(\frac{1}{n}) = 1 \tag{1}$$

$$(\frac{ab}{n}) = (\frac{a}{n})(\frac{b}{n}) \tag{2}$$

$$(\frac{a}{n}) = (\frac{a \bmod n}{n}) \tag{3}$$

With n odd

$$(\frac{-1}{n}) = (-1)^{\frac{n-1}{2}} \tag{4}$$

$$(\frac{2}{n}) = (-1)^{\frac{n^2-1}{8}} \tag{5}$$

With a and n odd and gcd(a,n) = 1

$$(\frac{-a}{n})(\frac{n}{a}) = (-1)^{\frac{(n-1)(a-1)}{4}} \tag{6}$$

4.2 If $a \leq 1$ then return a

$$\begin{aligned} a &= 1 \Rightarrow (\frac{1}{n}) = 1 = a && \textbf{(1)} \\ a &= 0 \Rightarrow (\frac{0}{n}) = 0 = a && \text{Definition of Legendre symbol} \end{aligned}$$

So we have to return a.

4.3 a is odd and $a \equiv n \equiv 3 \pmod 4$

With **(6)**, we have :

$$(\frac{a}{n}) = (\frac{n}{a}) \times (-1)^{\frac{(n-1)(a-1)}{4}}$$

As $a \equiv n \equiv 3 \pmod 4$, we have also :

$$\begin{cases} a = 4m+3 \\ n = 4p+3 \end{cases} \Rightarrow \begin{cases} a-1 = 4m+2 \\ n-1 = 4p+2 \end{cases}$$

$$\Rightarrow \frac{(a-1)(n-1)}{4} = \frac{(4m+2)(4p+2)}{4} = 2(2mp+p+m)+1$$

which is odd. So we obtain :

$$(\frac{a}{n}) = -(\frac{n}{a}) = -(\frac{n \bmod a}{a}) \qquad \textbf{(3)}$$

4.4 a is odd and ($a \equiv 1 \pmod 4$ or $n \equiv 1 \pmod 4$)

If $a \equiv 1 \pmod 4$ and $n \equiv 3 \pmod 4$, then :

$$\begin{cases} a-1 = 4m \\ n-1 = 4p+2 \end{cases} \Rightarrow \frac{(a-1)(n-1)}{4} = \frac{4}{4} \times (m(4p+2)) = 2(2mp+m)$$

which is even, so :

$$(\frac{a}{n}) = +(\frac{n \bmod a}{a})$$

The process would be exactly the same if ($a \equiv 3 \pmod 4$ and $n \equiv 1 \pmod 4$) or if ($a \equiv 1 \pmod 4$ and $n \equiv 1 \pmod 4$), we would obtain the same result :

$$(\frac{a}{n}) = +(\frac{n \bmod a}{a})$$

4.5 a is even and $n \equiv \pm 1 \pmod 8$

a is even so we can use **(2)** and then **(5)** :

$$(\frac{a}{n}) = (\frac{\frac{a}{2}}{n})(\frac{2}{n}) = (\frac{\frac{a}{2}}{n})(-1)^{\frac{(n^2-1)}{8}}$$

As $n \equiv \pm 1 \pmod 8$, we can deduct from it that :

$$\begin{cases} n = 8m+1 \\ n = 8p-1 \end{cases} \Rightarrow \begin{cases} n^2 = (8m+1)^2 = 64m^2+16m+1 = 8(8m^2+2m)+1 \\ n^2 = (8p-1)^2 = 64p^2-16p+1 = 8(8p^2-2p)+1 \end{cases}$$

$$\Rightarrow \begin{cases} \frac{n^2-1}{8} = 8m^2+2m = 2(4m^2+m) \\ \frac{n^2-1}{8} = 8p^2-2p = 2(4p^2-p) \end{cases}$$

So $\frac{n^2-1}{8}$ is always even and we can say that :

$$(\frac{a}{n}) = +(\frac{\frac{a}{2}}{n})$$

4.6 a is even and $n \not\equiv \pm 1 \pmod 8$

n is odd, $n \neq \pm 1 \pmod 8$, $n \neq 3 \pmod 8$ because we are in the case where $n \neq 3 \pmod 4$, so $n \equiv 5 \pmod 8$, therefore :

$$n = 8p + 5 \Rightarrow n^2 - 1 = (64p^2 + 80p + 25) - 1 = 8(2(4p^2 + 5p + 1) + 1)$$

$$\Rightarrow \frac{n^2 - 1}{8} = 2(4p^2 + 5p + 1) + 1$$

So $\frac{n^2-1}{8}$ is odd and as a consequence, we have :

$$(\frac{a}{n}) = -(\frac{\frac{a}{2}}{n})$$

To conclude, we can say that we have demonstrated the Jacobi algorithm using the 6 properties.

4.7 Size of $|a| + |n|$ after 2 recursions

When we compute Jacobi(a , n), we obtain 2 different results after one recursion : Jacobi($n \bmod a$, a) and Jacobi($\frac{a}{2}$, n) and their size are :

$$|n \bmod a| + |a| \text{ and } |\frac{a}{2}| + |n|$$

As in the second case, a is divided by 2, we won't develop this case anymore, which answers already the condition. After one more recursion of the first case, we obtain : Jacobi($a \bmod (n \bmod a)$, $n \bmod a$) and Jacobi($\frac{n \bmod a}{2}$, a) and their size are :

$$|a \bmod (n \bmod a)| + |n \bmod a| \text{ and } |\frac{n \bmod a}{2}| + |a|$$

As we did before, we will not study the second case because of the division by 2, which make us sure that the size of bits has decreased by at least one bit. $|a \bmod (n \bmod a)|$ is more interesting, indeed we have to prove that this term is lower than $\frac{a}{2}$. The only thing I manage to prove is that :

$$a \bmod (n \bmod a) \leq n \bmod a \leq a$$

5 Exercise 5 : MAPLE^{TM}

5.1 pqsqrt function

```
with(numtheory);
pqsqrt := proc(x,p,q)
        if (isprime(p) and isprime(q)) then
                return msqrt(x,p*q):
        else
                return "FAIL":
        end if;
end:
```

5.2 Picking p and q

```
p := 4:
q := 4:

while not isprime(p) and not isprime(q) do

        gen := rand(10^100..10^101-1):

        p := gen():
        q := gen():

        p := (p + 1260067572*10^101)*100 + 1:
        q := (q + 1260067572*10^101)*100 + 3:
end do:
n := p*q:
```

I obtain :

$$
\begin{aligned}
p &= 126006757279609549148208380785515299324819934583522895813335242694325685023233750873693355695318035853094102254013 \\
q &= 126006757226298343523655202549500646681772062200387990190406569558324008066172320097816330453818131658644985134030 \\
n &= 158777028734048619595607730880802169840325244289219771519262798978891709236044020764451759053004631939100715413815193567417720554891622575256004015622416251953826867272495062358474546321084979176747938718089833810851649549603
\end{aligned}
$$

5.3 Picking y and z

```
with(numtheory):
getTwoQNR := proc(n,y,z)
        y := rand();
        z := rand();

        while (jacobi(y,n) <> 1 and jacobi(z,n) <> -1) do

                y := rand();
                z := rand();

        end do;

        return y:
        return z:
end:
```

I do not understand how we can find a QNR element with a jacobi symbol equals to 1. Anyway, this code does not work : the loop does not stop when it should.

5.4 Square root of x, xy, xz, xyz

```
x := 1234567890:
```

```
i := 0:

while x <> 1234567990 do

        sqx := Vector([sqx,pqsqrt(x,p,q)]):
        sqxy := Vector([sqxy,pqsqrt(x*y,p,q)]):
        sqxz := Vector([sqxz,pqsqrt(x*z,p,q)]):
        sqxyz := Vector([sqxyz,pqsqrt(x*y*z,p,q)]):
        x := x+1:
end do;
```

I did not manage to store all the data along the iteration, so i cannot generate the required table.

6 Bibliography

6.1 Chinese Remainder theorem

- http://www.sciences-en-ligne.com/momo/chronomath/chrono1/Gauss.html
- http://www.les-mathematiques.net/b/a/d/node11.php3
- http://perso.club-internet.fr/orochoir/Maths/chinois.htm