Homework 1 - Cryptography and Security

Lucie Hoffmann

25.09.2020

Exercise 1

We define the following events:
A = 'me having the disease'
B = 'test is positive'.

Question 1.1

We know that $\Pr(B \mid A) = 0.99$, $\Pr(\overline{B} \mid A) = 0.01$, $\Pr(\overline{B} \mid \overline{A}) = 0.99$, $\Pr(B \mid \overline{A}) = 0.01$ and $\Pr(A) = 0.01$.

Then, $\Pr(A \mid B) = \frac{\Pr(B|A)\Pr(A)}{\Pr(B)} = \frac{\Pr(B|A)\Pr(A)}{\Pr(B|A)\Pr(A)+\Pr(B|\overline{A})\Pr(\overline{A})} = \frac{0.99\times0.01}{0.99\times0.01+0.01\times0.99}$ $= 0.5$

Question 1.2

Let a random variable X be the number of times the old test is positive knowing that the patient is not sick. It follows a Binomial distribution with parameters n and p = $\Pr(B \mid \overline{A}) = 0.01$. We want to know $\Pr(X \geq \frac{n}{2}) \leq \frac{\mathrm{E}[X]}{\frac{n}{2}} = \frac{2np}{n} =$ 2p = 0.02. The more old tests we make, the more likely the test's result will be correct. Hence the bigger n, the lower the probability of a false positive. The bound we found does not decrease as n grows so it is not tight.

Exercise 2

Question 2.1

1. Defining d as the degree of a polynomial in $\mathcal{R} \setminus \{0\}$, we verify that the 3 properties hold.
Let a and b be in $\mathcal{R} \setminus \{0\}$ such that $d(a) = n$ and $d(b) = k$.
We write $a = a_n x^n + ... + a_0$ and $b = b_k x^k + ... + b_0$, where $a_n, ..., a_0$ and $b_k, ..., b_0$ are coefficients in K, in particular, a_n and b_k are non-zero.
If $k > n$, then we take $m = 0$ and $r = a$. If $k \leq n$, then we have $a = (b_m x^m + ... + b_0)\frac{a_n}{b_m} x^{n-m} + p$,

where $p = 0$ if $b \mid a$, $p = (a_{n-1} - \frac{a_n b_{n-1}}{b_n})x^{n-1} + ... + (a_0 - \frac{a_n b_0}{b_n}) = a - b\frac{a_n}{b_k}x^{n-k}$ if $b \nmid a$.
We find $r = a - b(\frac{a_n}{b_k}x^{n-k} + ...)$ by repeating the above subtraction with p, until $d(r) < d(b)$. Since the degree is strictly decreasing, this process terminates.
Next, since K is a field, $a_n b_m$ is non-zero and $d(a) = n, d(b) = m \leq d(ab) = n + m$. Finally, since $1 \in \mathcal{R}$, there exists $x^l \in \mathcal{R}$ such that ≥ 1, i.e. $d(x^l) > 0$.
Hence $\mathcal{R}$ is a Euclidean domain.

2. We prove by contradiction that $K[x_1, x_2]$ is not a Euclidean domain.
Suppose $K[x_1, x_2]$ is a Euclidean domain. This implies it is a principle ideal domain, i.e. every ideal is generated by a single element (proved in Q2.2). But this is a contradiction since $K[x_1, x_2]$ is a bivariate polynomial ring, i.e. it is generated by two elements x_1, x_2.
Hence the ideal $K[x_1, x_2]$ is not generated by a single element and $K[x_1, x_2]$ is not a Euclidean domain.

Question 2.2

Let the ideal $I \in \mathcal{R}$ such that $I \neq \{0\}$. Indeed the ideal $\{0\}$ is generated by 0. We take $a, x \in I$ such that $d(x)$ is minimum in I. Then there exist $m, r \in \mathcal{R}$ such that $a = xm + r$ with either $r = 0$ or $d(r) < d(x)$. Since we know $d(r) \geq d(x)$ $\forall r \in I$, we have $r = 0$. This implies $\forall a \in I, \exists m \in \mathcal{R}$ such that $a = xm$ and $I = x\mathcal{R}$, i.e. every ideal in $\mathcal{R}$ is generated by a single element (=principal).

Question 2.3

1. Let an ideal $I \subseteq \mathbb{Z}$, $a, b \in I$ and $c \in \mathbb{Z}$ such that $a\mathbb{Z} + b\mathbb{Z} = c\mathbb{Z}$. This implies there must exist $u, v \in \mathbb{Z}$ such that $au + bv = c$. This is Bezout identity where c is the GCD of a and b. On the other hand, if we take $a, b, c \in \mathbb{Z}$ such that c is the GCD of a and b, i.e. there exist $u, v \in \mathbb{Z}$ such that $au + bv = c$, then $aw + bx = cy$ is verified for every $w, x, y \in \mathbb{Z}$ since only need to multiply both sides of the equation by y. Hence both definitions of the GCD are compatible.

2. At each step of the Euclidean algorithm, for $a, b \in \mathcal{R}$, we want to find $m, r \in \mathcal{R}$ such that $a = mb + r$ where either $r = 0$ or $d(r) < d(b)$. When $r = 0$ then the algorithm terminates and the GCD is the last such b. Since $\mathcal{R}$ is a Euclidean domain, we know that there exist such m and r at each step. Hence we can replace the comparison of two integer in $\mathbb{Z}$ by the comparison of two image of elements in $\mathcal{R}$ w.r.t. the d function. In the end, when looking for GCD(a, b), we want to find the generator c of the ideal $I = a\mathcal{R} + b\mathcal{R} = c\mathcal{R}$, i.e. such that $d(c)$ is minimum in I.

Exercise 3

Question 3.1

We show by induction that the given assertion, call it $P(n)$ holds for all $n = 2^k$ where $k \in \mathbb{N}$.
Basis step:
For $k = 0$, $n = 1$ and

$$T(1) = d = \begin{cases} d + S(1)\log(1) = d & \text{if } b = c \\ d + \frac{c}{b-c}S(1) \times 0 = d & \text{if } b \neq c \end{cases}$$

We assume $P(n)$ is true for some n and k, we prove this implies $P(n+1)$ is also true.
Inductive step:

$$T(n+1) \leq bT(\frac{n+1}{2}) + S(n+1) \tag{1}$$

$$\leq \begin{cases} b(d(\frac{n+1}{2})^{\log(b)} + S(\frac{n+1}{2})\log(\frac{n+1}{2})) + S(n+1) & \text{if } b = c \\ b(d(\frac{n+1}{2})^{\log(b)} + \frac{c}{b-c}S(\frac{n+1}{2}((\frac{n+1}{2})^{\log(\frac{b}{c})} - 1)) & \text{if } b \neq c \end{cases} \tag{2}$$

$$\leq \begin{cases} d(n+1)^{\log(b)} + S(n+1)(\log(n+1) - 1) + S(n+1) \\ d(n+1)^{\log(b)} + \frac{b}{b-c}S(n+1)(\frac{(n+1)^{\log(\frac{b}{c})}}{\frac{b}{c}} - 1) \end{cases} \tag{3}$$

$$= \begin{cases} d(n+1)^{\log(b)} + S(n+1)\log(n+1) \\ d(n+1)^{\log(b)} + \frac{c}{b-c}S(n+1)((n+1)^{\log(\frac{b}{c})} - 1) \end{cases}$$

Where in (1) we used the second inequality from the statement, in (2) we used the induction hypothesis, in (3) we used the first inequality from the statement. By the induction principle, we conclude that $P(n)$ is true for all $n = 2^k$ where $k \in \mathbb{N}$.

Question 3.2

Case where $b = c$:
We observe that $T(2) \leq bd + S(2) \leq AS(2)\log(2)$ if we take $A \geq 1 + \frac{bd}{S(2)}$. We assume $T(N) \leq AS(N)\log(n)$ for some N and $A > 0$. For $n \geq N$, we have:

$$T(n) \leq bT(\frac{n}{2}) + S(n)$$

$$\leq bAS(n/2)\log(\frac{n}{2}) + S(n) \tag{1}$$

$$\leq \frac{b}{c}AS(n)(\log(n) - 1) + S(n) \tag{2}$$

$$= AS(n)\log(n) + S(n)(1 - A)$$

where in (1) we used the induction hypothesis, in (2) the statement $S(2n) \geq cS(n)$.
Choosing $A \geq \max(1 + \frac{bd}{S(2)}, d, 1)$ satisfies $T(n) \leq AS(n)\log(n)$ both in the base and inductive cases. Hence $T(n) \in O(S(n)\log(n))$ when $b = c$.

Case where $b \neq c$:
We observe that $T(1) = d \leq AS(1) \times 1 - A'S(1)$ if $A - A' \geq d$. We assume $T(N) \leq AS(N)N^{\log(\frac{b}{c})} - A'S(N)$ for some N and $A, A' > 0$. For $n \geq N$, we have:

$$T(n) \leq bAS(\frac{n}{2})(\frac{n}{2})^{\log(\frac{b}{c})} - bA'S(\frac{n}{2}) + S(n) \tag{3}$$

$$\leq bA2^{\log(\frac{c}{b})}cS(n)n^{\log(\frac{b}{c})} - bA'\epsilon S(n) + S(n) \tag{4}$$

$$= AS(n)n^{\log(\frac{b}{c})} + (1 - bA'\epsilon)$$

$$\leq AS(n)n^{\log(\frac{b}{c})} - A'S(n) \tag{5}$$

where in (3) we start with the statement's inequality on $T(n)$ and use the induction hypothesis, in (4) we use $S(2n) \geq cS(n)$ and $S(2n) \leq \epsilon S(n)$, in (5) we set $A' \geq \frac{1}{b\epsilon - 1}$. We have found conditions on A and A' that satisfies both base and inductive cases. Hence $T(n) \in O(S(n)n^{\log(\frac{b}{c})})$ for $b \neq c$.

Question 3.3

1. For $n = 1$, f and g are constants and $h = fg$. We assume the algorithm is correct for some $n = 2^k$. We prove this holds for $m = n + 1$:
We have $h = F_1G_1x^m + ((F_0 + F_1)(G_0 + G_1) - F_1G_1 - F_0G_0)x^{\frac{m}{2}} + F_0G_0 = F_1G_1x^m + (F_1G_0 + F_0G_1)x^{\frac{m}{2}} + F_0G_0 = (F_1x^{\frac{n}{2}} + F_0)(G_1x^{\frac{n}{2}} + G_0) = fg$. By the induction principle, the algorithm indeed outputs the multiplication of the two polynomials f and g.

2. For $n = 1$, Karatsuba's algorithm needs one ring operation: $T(1) = 1 = d$. For $n > 1$, we recursively call Karatsuba's algorithm with at most $\frac{n}{2}$ and we compute h with at most $4n$ ring operations. We thus have $T(n) \leq 3T(\frac{n}{2}) + 4n$. We use the previous results (first case in (1)) by observing that $b = 3$, $S(n) = 4n$ and $S(2n) \geq 2S(n)$ with $c = 2$:
$T(n) \leq n^{\log(3)} + 2 \times 4n(n^{\log(\frac{3}{2})} - 1) = n^{\log(3)} + 8n^{1+\log(\frac{3}{2})} - 8n = 9n^{\log(3)} - 8n$.

3. $\log(3) = 1.59$ and $9n^{1.59} - 8n \leq 9n^{1.59}$ hence the number of ring operations required is $O(n^{1.59})$.