Exercise Worksheet 2

November 6, 2022

1 Exercise 1

A *Euclidean domain* is a ring in which the Euclidean algorithm can be applied (for example, the integers). Let R be a Euclidean domain, K its field of fractions, and $f_1, \ldots, f_l \in K$.

A *continued fraction*, denoted $C(f_1, \ldots, f_l)$, is defined to be

$$f_1 + \cfrac{1}{f_2 + \cfrac{1}{\ddots \, \cfrac{}{f_{l-1} + \cfrac{1}{f_l}}}}.$$

Let $\{ q_i : 1 \le i \le l \}$ be the quotients in the Euclidean algorithm run on $r_0, r_1 \in R$.

1.1 Part (a)

Prove by induction that $r_0/r_1 = C(q_1, \ldots, q_l)$.

1.2 Part (b)

Representing a continued fraction as a list $[q_1, \ldots, q_l]$, write a Sage or Maple procedure `contfrac` to compute the continued fraction expansion of two polynomials in $\mathbb{Q}[x]$.

1.3 Part (c)

Run your algorithm on $r_0 := x^{20}$ and $r_1 := x^{19} + 2x^{18} + x \in \mathbb{Q}[x]$.

2 Exercise 2

This exercise considers a variant of the Euclidean algorithm that can be faster in practice. Consider the following recursive pseudocode for computing $\gcd(a, b)$ of two positive integers a and b.

- if $a = b$ then return a
- if both a and b are even then return $2\gcd(a/2, b/2)$
- if a is even then return $\gcd(a/2, b)$
- if b is even then return $\gcd(a, b/2)$
- if $a > b$ then return $\gcd((a-b)/2, b)$
- otherwise return $\gcd((b-a)/2, a)$

2.1 Part (a)

Implement this algorithm in Sage or Maple and demonstrate it on the pairs $(34, 21)$, $(136, 51)$, $(481, 325)$, and $(8771, 3206)$.

2.2 Part (b)

Use induction to prove the algorithm works correctly. (Hint: use *strong induction* which derives a proposition about a number by assuming the proposition is true for all smaller numbers.)

2.3 Part (c)

Find a good upper bound on the recursion depth and use this to prove that the running time of the algorithm is $O(n^2)$ word operations when a and b have length at most n.

2.4 Part (d)

Modify the algorithm into an "extended" version which computes integers s, t such that $sa + tb = \gcd(a, b)$. Give your answer in the form of a Sage or Maple function and test it on the pairs from part (a).

3 Exercise 3

If p is a prime then the ring $\mathbb{Z}_p$ of integers mod p is a field: every nonzero element has an inverse.

In particular, for any nonzero $b \in \mathbb{Z}$ and any $a \in \mathbb{Z}$ we can always find a $q \in \mathbb{Z}$ such that $a \equiv qb \pmod{p}$.

When m is not prime the congruence $a \equiv qb \pmod{m}$ may or may not have a solution. For example, $6 \equiv 3 \cdot 6 \pmod{12}$ (so with $a = b = 6$ and $m = 12$ there is a solution) but there is no integer q such that $5 \equiv q \cdot 6 \pmod{12}$ (so with $a = 5$, $b = 6$, $m = 12$ there is no solution).

3.1 Part (a)

Write a Sage or Maple function `mod_inv` that takes as input integers a and b and returns some element $q \in \mathbb{Z}$ such that $a \equiv qb \pmod{m}$ or returns False if no such q exists. You can use the the `xgcd` function of Sage or the `igcdex` function of Maple. Your implementation should run in polynomial time in the input size.

3.2 Part (b)

Test your function well and demonstrate it working on several different inputs.

4 Exercise 4

Let $q = 11$ and $n = 10$. This question will involve Reed–Solomon codes over $\mathbb{F}_q$.

4.1 Part (a)

Show that $\alpha = 2 \in \mathbb{F}_q$ is a primitive nth root of unity and that the polynomial $x^n - 1$ splits into linear factors over $\mathbb{F}_q$.

4.2 Part (b)

Suppose that we want to correct up to $t = 2$ errors. Show that $g(y) = y^4 + 3y^3 + 5y^2 + 8y + 1$ works as a generator polynomial.

4.3 Part (c)

Suppose that you receive the encoded message $y^6 + 7y + 4$. What is the corrected codeword and what was the original message?