The bit-complexity of multiplication

Multiplication: $O(n^{\log_2 3})$

Theorem (Schönhage & Strassen 1971)

The product of two n-bit numbers can be computed in time $O(n \log n \log\log n)$.

Oberwolfach.

Famous open problem:

$O(n \cdot \log n)$?

The bit-complexity of multiplication

Theorem (Fürer 2009)

Two n-bit integers can be computed in time $O(n \log n 2^{O(\log^* n)})$.

$\log^* n$:

log-operations needed to bring n down to 1.

(base 2)

Division with remainder

Theorem

For $a, b \in \mathbb{N}$, $b \neq 0$, there exists $q, r \in \mathbb{N}$ with

$q \in \mathbb{Z}$

i) $a = q \cdot b + r$

ii) $0 \leq r < b$.

Division with remainder.

$12 = 2 \cdot 5 + 2$

$q \quad q \quad r$

$q.$

Question: How fast can we compute q and r on input a and b?

Algorithm

```
function divide(a,b)   # b>= 1

if a < b: return (q,r)=(0,a)

(q,r) = divide([a/2],b)
q = 2q, r = 2r
if a is odd:
    r = r+1
if r>=b:
    r = r-b
    q = q+1
return (q,r)
```

Assume.
$a, b \in \mathbb{N}$, $b \geq 1$

$\lfloor a/2 \rfloor = q \cdot b + r \quad | * 2.$

$\lfloor a/2 \rfloor \cdot 2 = (2 \cdot q) \cdot b + 2 \cdot r$

$r \leq 2 \cdot b - 1$

O(1) ← # of bits of q grows by 1 !

$r \leq 2 \cdot b$

O(size(b))

$r < b$

O(1) ← update q.

All together $O(\text{size}(q) \cdot \text{size}(b))$

Analysis

Theorem

Let $a, b \in \mathbb{N}$, $b \geq 1$. The algorithm `divide` *runs in time* $O(\text{size}(q) \cdot \text{size}(b))$ *on Input a, b.*

Remark: Result of Schönhage & Strassen later with technique based on Newton iteration implies that D.w.R. can be done in time $O(n \log n \log\log n)$, (Fürer: $O(n \log n \, 2^{\log^* n})$)

n : # Bits in total of input.

The greatest common divisor

- $a, b \in \mathbb{Z}$ not both equal to zero.
- $\gcd(a, b) = \max\{d \in \mathbb{N} : d \mid a,\ d \mid b\}$.

q
divides

gcd (-15, -9) = 3

The Euclidean algorithm

Lemma

Let $a, b \in \mathbb{N}$ $b \neq 0$ and let $q, r \in \mathbb{N}$ with

$$a = q \cdot b + r, \quad 0 \leq r < b,$$

then $\gcd(a, b) = \gcd(b, r)$.

Suppose $d \mid a$ and $d \mid b$. $\exists x_a, x_b \in \mathbb{Z}$ s.t. $d \cdot x_a = a$, $d \cdot x_b = b$.

$r = a - q \cdot b = d(\underbrace{x_a - q \cdot x_b}_{\in \mathbb{Z}}) \Rightarrow d \mid r.$

Similarly: $d \mid b$ and $d \mid r \Rightarrow d \mid a$

$\Rightarrow \gcd(a, b) = \gcd(b, r).$

The Euclidean algorithm

How many iterations?:

Observation: $r \leq a/2$


```
function gcd(a, b)

Input: a, b ∈ ℕ0 with a ≥ b and a > 0
Output: gcd(a, b)

if b = 0 return a
else
    Compute q, r ∈ ℕ with a = q · b + r, 0 ≤ r < b.
    return gcd(b, r)
```

Suppose $r \geq a/2$

$a = q \cdot b + r \geq 1 \cdot (a/2 + 1) + a/2 > a$ ↯.

Since $a \geq b$ and $r < b$ $\Rightarrow q \geq 1$.

Theorem: The number of iterations of gcd(a,b) is $O(\text{size}(a))$

Naive upper bound on # of bit-operations:

$O(\text{size}(a) \cdot \text{size}(a) \cdot \text{size}(b)) \leq O(\text{size}(a)^3)$

Analysis: Lower bound

$\gcd(F_N, F_{N-1})$

Theorem

Suppose a and b are two n-bit integers. The Euclidean algorithm requires $\Omega(n^2)$ time, regardless of the algorithms used for the arithmetic operations.

Proof: Remember $F_0 = 0$, $F_1 = 1$, $F_N = F_{N-1} + F_{N-2}$ $N \geq 2$.

$F_n = q \cdot F_{N-1} + r$

$q \overset{?}{=} 1$

$r \overset{?}{=} F_{N-2}$

Euclidean Alg writes down complete sequence of Fibonacci Numbers. $F_N \geq 2 \cdot F_{N-2} \geq \cdots \geq 2^{\lfloor N/2 \rfloor}$

$F_N = \Theta(N)$ bits.

$\Rightarrow \Theta(N) + \Theta(N-1) + \cdots + \Theta(1) = \Theta(N^2)$

Analysis: Upper bound

Theorem

Suppose a and b are two n-bit integers. If the previously described algorithm for division with remainder is used, then the Euclidean algorithm requires $O(n^2)$ time.

Proof: We show by Induction: There exists a constant c s.t. $gcd(a,b)$ $a \geq b \geq 1$ requires at most $c \cdot \log(a+1) \cdot \log(b+1)$ bit operations.

$= \Theta(\text{size}(a))$ $\Theta(\text{size}(b))$

Induction on # of recursive calls. T.

$T=1$ straight forward. $O(\text{size}(a) \cdot \text{size}(b))$

$T \to$: First division with remainder : $c \cdot \log(q+1) \cdot \log(b+1)$ steps.

Induction: rest $c \cdot \log(b+1) \cdot \log(r+1)$ $\in \log(q \cdot b + r + 1) = \log(a+1)$

Together: $c \cdot \log(b+1) \left[\log(q+1) + \log(r+1) \right] = c \cdot \log(b+1) \log(q \cdot r + r + q + 1)$ ∎

The extended Euclidean Algorithm

function exgcd(a, b)

Input: $a, b \in \mathbb{N}_0$ with $a \geq b$ and $a > 0$

Output: $(\gcd(a, b), x, y)$ with

if $b = 0$ return $(a, 1, 0)$

else

Compute $q, r \in \mathbb{N}$ with $a = q \cdot b + r, 0 \leq r < b$.

$(d, x', y') = \text{exgcd}(b, r)$

return $(d, y', x' - q \cdot y')$

$x' \cdot b + y' \cdot r = d$

plug in $r = a - q \cdot b$

a, b not both 0.

Thm: $\gcd(a,b) = \min\{x \cdot a + y \cdot b : x, y \in \mathbb{Z}, x \cdot a + y \cdot b \geq 1\}$

proof: „$\geq$" $d \in \mathbb{N}$, common divisor of a, b.

$\Rightarrow \exists x_a, x_b \in \mathbb{N}$ with $a = x_a \cdot d$, $b = x_b \cdot d$

$1 \leq \underbrace{x \cdot a + y \cdot b}_{=\min} = \underbrace{(x \cdot x_a + y \cdot x_b)}_{\in \mathbb{Z}} \cdot d$

$a = 1 \cdot a + 0 \cdot b$ (✓)

$\Rightarrow d \mid \min \Rightarrow \gcd \geq \min$

„$\leq$" To show: $\min \mid a$ and b.

Suppose not: suppose $\min \nmid a$.

then $a = q \cdot \min + r$, $1 \leq r < \min$

$\Rightarrow r = a - q \cdot \min = a - q(x \cdot a + y \cdot b)$

$= (1 - qx)a - qy\,b$

to minimality

Analysis

Theorem

The extended Euclidean algorithm runs in time $O(\text{size}(a) \cdot \text{size}(b))$

Exercise: Key observation $|x| \leq \frac{b}{\gcd(a,b)}$, $|y| \leq \frac{a}{\gcd(a,b)}$

Analysis very similar to previous analysis!

Equivalence relations

Definition

A binary relation $\sim$ on a set X is an *equivalence relation* if for all $x, y, z \in X$ one has:

I) (Symmetry) If $x \sim y$, then $y \sim x$

II) (Reflexivity) $x \sim x$

III) (Transitivity) If $x \sim y$ and $y \sim z$, then $x \sim z$.

Example

Let $N \neq 0$ be an integer and $X = \mathbb{Z}$. The relation

$$x \sim y \text{ if } N \mid (x - y)$$

is an equivalence relation.

i) $N \mid (x-y) \Leftrightarrow N \mid (y-x)$

ii) $N \mid 0$

iii) $N \mid (x-y)$ and $N \mid (y-z)$

$\Rightarrow N \mid (x-y)+(y-z) = (x-z)$

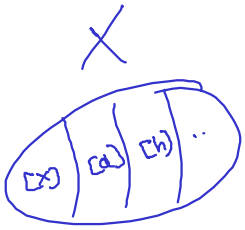
Equivalence class

Definition

Let $\sim$ be an equivalence relation on X and $x \in X$. Then

$$[x] = \{y \in X : x \sim y\}$$

is the *equivalence class* of x.



Lemma

Let $[x]$ and $[y]$ be two equivalence classes. If they are not disjoint, then they are equal.

$z \in [x] \cap [y]$, then $x \sim z$, $y \sim z$ $\overset{\text{trans.}}{\Rightarrow}$ $x \sim y$

Suppose that $h \in [x]$, then $h \sim x$, $x \sim y$

$\Rightarrow h \sim y$, $h \in [y]$

$\Rightarrow [x] \subseteq [y]$ sim. $[x] \supseteq [y]$. ∎

Definition

Let N be a nonzero integer. The *integers modulo* N is the set of equivalence classes of the equivalence relation $x \sim y$ if $N \mid (x - y)$.

Example: $N = 3$ $\quad [1] = \{\dots, -5, -2, 1, 4, 7, \dots\}$

$[1] = [-5] \in \mathbb{Z}_N$.

1 is representative of the equivalence class to which it belongs.

Groups

Definition

A *group* consists of a set G and a binary operation $\odot$ that takes two group elements $a, b \in G$ and maps them to another group element $a \odot b \in G$ such that the following conditions hold.

a) (Associativity) For all $a, b, c \in G$ one has $(a \odot b) \odot c = a \odot (b \odot c)$.

b) (Neutral element) There exists an element $e \in G$ with $a \odot e = e \odot a = a$ for all $a \in G$.

c) (Inverse Element) For each $a \in G$ there exists a $b \in G$ with $a \odot b = b \odot a = e$.

If $a \odot b = b \odot a$ for all $a, b \in G$, then the group is *commutative* or *abelian*.

↑ Neutral element!

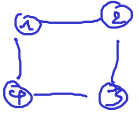
Example

1. The integers with $\odot = +$ are a group.

$\odot = \cdot$ No! inverse element does not exist.

$2 \cdot x \overset{!}{=} 1$

Example:

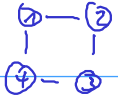


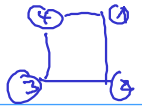
0°

90°

180°

270°





3 4
2 1

2 3
1 4

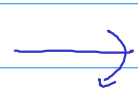
Rotations

Reflections:

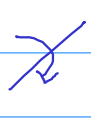
$O_{p_1} \circ O_{p_2} \circ \cdots \circ$

$O_{p_1} \circ O_{p_2} \overset{?}{=} O_{p_2} \circ O_{p_1}$

No.



2 1
3 4



1 4
1 3

$(\mathbb{Z}_N, \oplus)$

Lemma

The binary operation

$$[a] \oplus [b] = [a + b]$$

is well defined. No matter which representative a, b of $[a]$ and $[b]$ is chosen, the result $[a+b]$ is always the same.

Lemma

$\mathbb{Z}_N$ *together with* $\oplus$ *is an abelian group.*

Defin $\odot$ for $\mathbb{Z}_N \setminus \{[0]\}$.

$[a] \odot [b] = [a \cdot b]$ also this is well-defined.

Question: is $(\mathbb{Z}_N^{\setminus\{[0]\}}, \odot)$ a group?

No: $N = 4$, $[2]$ does not have a multiplicative inverse: $4 \mid (2 \cdot X - 1)$ is not possible, since this implies $2 \mid (2 \cdot X - 1) \Rightarrow 2 \mid 1$ ✗ Not sense.

$(\mathbb{Z}_N, \odot)$ is not a group if N is not a Prime number.

Theorem: $[a] \in \mathbb{Z}_N$ has multiplicative inverse

$\Leftrightarrow \gcd(a, N) = 1$.

Proof: Suppose $\gcd(a, N) = d \geq 2$.

$[a][x] = [1]$

if $N \mid (a \cdot x - 1) \Rightarrow d \mid a \cdot x - 1$

$\Rightarrow d \mid 1$ ↯.

$\Rightarrow [d]$ has no mult. inverse.

if $\gcd(a, N) = 1 \Rightarrow \exists x, y \in \mathbb{Z} : x \cdot a + y \cdot N = 1$.

$[x][a] = [xa] = [1 - y \cdot N] = [1]$ ∎