Computer Algebra

Sebastian Iwanowski
FH Wedel

2. Integer Arithmetics
2.1 Dividing integer numbers, rational arithmetic

References for repetition and deepening your knowledge:

Köpf 3.3, 3.6, Kaplan 4.1.4, 4.1.5, 4.2 (in German)
Knuth 4.5.1 – 4.5.3 (vol. 2) (Euclidean algorithm)

Computer Algebra 2

Algorithm for dividing long numbers a and b

- Division with remainder (school method) (for base β=10)

DIVIDE (I: $[i_{n-1}\ i_{n-2}\ \dots\ i_0]$, J: $[j_{m-1}\ j_{m-2}\ \dots\ j_0]$): $[q_{l-1}, q_{l-2}, \dots\ q_0]$ run time: $O(n^2)$

```
J > I => return [0];
Q := [ ];              /* initialisation of result with empty list */
I* := [i_{n-1} i_{n-2} … i_{n-m}] ;  /* new dividend: will be expanded digit by digit in the following */
f:= n-1; k := n-m      /* first and last index of I* within input dividend I */
while (k ≥ 0) do
  { if I* < J
       { append (Q, 0); }
    else
       { if (length(I*) > length(J))
            { qTest := (i_f •10 + i_{f-1}) DIV j_{m-1}; /* short number operations */
              if (qTest > 9) {qTest := 9;} /* higher digits are not feasible */  }
         else
            { qTest := i_f DIV j_{m-1};}; /* short number division */
       J* := qTest • J;
       while (J* > I*) do
          { qTest := qTest-1; J* := qTest • J; }
       append (Q, qTest);
        I* := I* - J*; f := index of first digit of I* not equal to 0 or k-1 if I*=0; }      /* end if */
    k := k-1; if (k ≥ 0) {I* := I*•10 + i_k; } /* expanding I* by one digit */ } /* end while */
return Q;
```

Theorem (Pope-Stein):
If j_{m-1} is at least β/2,
qTest exceeds the true value by at most 2.

Computer Algebra 2

Algorithm for dividing long numbers a and b

Let the size of both operands be O(n)

- Division with remainder

DIVIDE:
Estimation of integer quotient by Pope-Stein
(concerning run time, only the constant is improved)

run time: $O(n^2)$

also possible in $O(n^{\log_2(3)})$

more details (in German):
Kaplan, S. 74-79 (commented with patches)

MODULUS:
x MODULUS y = x – x DIVIDE y

run time: O(n)
(having computed the result of DIVIDE already)

DIVIDE works for positive operands only!

Computer Algebra 2

Rational Arithmetic

Operations for fractions

Let the size of numerator and denominator be $O(n)$

- Simplifying fractions: Euclidean algorithm

overall run time: $O(n^2)$

Proof is difficult, cf. Knuth

Theorem: Let $n = q \cdot m + r$ for integer numbers n,m,q,r, $0 \le r < m$

Then the following holds: gcd (n,m) = gcd (m,r)

Algorithm for n,m>0:

1) Compute q and r for n and m using DIVIDE and MODULUS of last slide

single run time: $O(n^2)$
(becomes less in progress)

2) If $r = 0$,
   then return gcd = m
   else replace $n := m$ und $m := r$ and continue at 1)

Easiest modification for negative n or m ?

Computer Algebra 2

Rational Arithmetic

Operations for fractions

Let the size of numerator and denominator be O(n)

- Operation rules

  just as in school math

  maximal run time: $O(n^{\log_2(3)})$

- Simplifying

  Apply Euclidean algorithm

  run time: $O(n^2)$

Conclusion: All rational operations are possible in $O(n^2)$