Long division of positive integers

Gábor P. Nagy

Budapest University of Technology and Economics (Hungary)

May 4, 2021

Outline

Outline

Hungarian (German) long division

$$1111 : 13 =$$

Hungarian (German) long division

$$1111 : 13 = 8$$

Hungarian (German) long division

$$
\begin{array}{l}
111'1 : 13 = 8 \\
\underline{-\;104} \\
\quad\;\;\; 7
\end{array}
$$

Hungarian (German) long division

$$111'1' : 13 = 85$$

$$\begin{array}{r} -104 \\ \hline 71 \end{array}$$

Hungarian (German) long division

$$
\begin{array}{rl}
1111 & :13=85\\
\underline{-104} & \\
71 & \\
\underline{-65} & \\
6 &
\end{array}
$$

Hungarian (German) long division

$$1111 : 13 = 85$$

$$\begin{array}{r} -104 \\ \hline 71 \\ -65 \\ \hline 6 \end{array}$$

quotient

rest

1111 divided by 13 is 85 with remainder 6

$$
\begin{array}{r}
85 \\
\hline
13)1111 \\
\underline{104} \\
71 \\
\underline{65} \\
\mathbf{6}
\end{array}
$$

Long division bottleneck

MAGIC!!

Figure out the digits 8 and 5.

EASY: $8 = 111/13$ with rest $7 = 111 - 8 \times 13$.
EASY: $5 = 71/13$ with rest $6 = 71 - 5 \times 13$.

PROBLEM: This may need 3-digits arithmetic!!
IN GENERAL, with n-digit divisor, $n + 1$-digit arithmetic is needed.

Division with a one-digit number

```
t_NAT t_NAT::divide_by_digit(t_digit a) const {
  if (!a) {
    std::cerr << "Divison␣by␣zero!" << std::endl;
    exit(EXIT_FAILURE);
  }
  t_digit rem = 0;
  t_NAT n = *this;
  for (auto i = n.arr.rbegin(); i != n.arr.rend(); ++i) {
    int b = RADIX * rem + (*i);
    *i = t_digit(b / a);
    rem = t_digit(b % a);
  }
  // cleaning leading zeros of n ...
  return n;
}
```

Base b representation recap

Generic n-digit integer in base b:

$$
\begin{aligned}
a &= (a_{n-1}a_{n-2}\dots a_1a_0)_b \\
&= a_{n-1}b^{n-1} + \cdots + a_1b + a_0 \qquad (0 \le a_i \le b-1) \\
\left\lfloor \frac{a}{b} \right\rfloor &= a_{n-1}b^{n-2} + \cdots + a_1 \\
&= (a_{n-1}a_{n-2}\dots a_1)_b \\
b^n &= (10\dots 0)_b \\
&= \text{smallest } n+1\text{-digit integer} \\
&> a \\
&\ge a_{n-1}b^{n-1} = (a_{n-1}0\dots 0)_b
\end{aligned}
$$

Outline

Dividing an $n+1$-digit number by an n-digit number

Problem 1

Given positive integers u, v in base b:

$$u = (u_n u_{n-1} \dots u_0)_b$$
$$v = (v_{n-1} v_{n-2} \dots v_0)_b$$

We assume $v_{n-1} \neq 0$ and $u/v < b$.
Find an algorithm to determine $q = \lfloor u/v \rfloor$ *(rapidly, even for large b).*

- $u/v < b \iff u/b < v \iff \lfloor u/b \rfloor < v \iff (u_n u_{n-1} \dots u_1)_b < (v_{n-1} v_{n-2} \dots v_0)_b$
- If $r = u - qv$, then q is the unique integer such that $0 \leq r < v$.

Obvious approach

Make a guess about q, based on the two most significant digits of u and v.

SPOILER: Good idea, except when v_{n-1} is small.

Make a guess about $q = \lfloor u/v \rfloor$

Problem 1

Given positive integers u, v in base b:

$$u = (u_n u_{n-1} \dots u_0)_b$$
$$v = (v_{n-1} v_{n-2} \dots v_0)_b$$

We assume $v_{n-1} \neq 0$ and $u/v < b$.
Find an algorithm to determine $q = \lfloor u/v \rfloor$ *(rapidly, even for large b).*

Define $\quad \hat{q} = \left\lfloor \dfrac{u_n b + u_{n-1}}{v_{n-1}} \right\rfloor. \quad$ If $\hat{q} \geq b$ then set $\quad \hat{q} = b - 1.$

In the previous example, $u = 111$, $v = 13$,

$$u_n b + u_{n-1} = 11, \quad v_{n-1} = 1, \quad \lfloor 11/1 \rfloor = 11,$$

hence $\hat{q} = 10 - 1 = 9.$

Lower bound on $\hat{q}$

Theorem A

In the notation $\hat{q} = \min\left(\left\lfloor \frac{u_n b + u_{n-1}}{v_{n-1}} \right\rfloor, b-1\right)$, we have $\hat{q} \geq q$.

Proof. Certainly true if $\hat{q} = b - 1$. Otherwise

$$\frac{u_n b + u_{n-1}}{v_{n-1}} - \left\lfloor \frac{u_n b + u_{n-1}}{v_{n-1}} \right\rfloor \leq 1 - \frac{1}{v_{n-1}} \implies \hat{q} v_{n-1} \geq u_n b + u_{n-1} - v_{n-1} + 1.$$

From $b^{n-1} > u_{n-2} b^{n-2} + \cdots + u_0 = (u_{n-2} \ldots u_0)_b$ follows

$$\begin{aligned} u - \hat{q} v &\leq u - \hat{q} v_{n-1} b^{n-1} \\ &\leq u_n b^n + \cdots + u_0 - (u_n b^n + u_{n-1} b^{n-1} - v_{n-1} b^{n-1} + b^{n-1}) \\ &= u_{n-2} b^{n-2} + \cdots + u_0 - b^{n-1} + v_{n-1} b^{n-1} < v_{n-1} b^{n-1} \leq v. \end{aligned}$$

Since $u - \hat{q} v < v$, we must have $\hat{q} > u/v - 1$.
$u/v - 1 \geq \lfloor u/v \rfloor - 1 = q - 1$ implies $\hat{q} > q - 1$. $\square$

A lemma

Lemma

1. If $v = b^{n-1} = (10\dots0)_b$, then $q = \hat{q}$.
2. If $v \neq b^{n-1}$ then
$$\hat{q} < \frac{u}{v - b^{n-1}}.$$

Proof. (1) $q = \lfloor u/v \rfloor = \lfloor u_n b + u_{n-1} + (\cdots)/b^{n-1} \rfloor = \lfloor u_n b + u_{n-1} \rfloor = \hat{q}$.
(2) We have

$$\hat{q} \leq \frac{u_n b + u_{n-1}}{v_{n-1}} = \frac{u_n b^n + u_{n-1} b^{n-1}}{v_{n-1} b^{n-1}} \leq \frac{u}{v_{n-1} b^{n-1}} < \frac{u}{v - b^{n-1}}.$$

The **last step** follows from

$$v - v_{n-1} b^{n-1} = (v_{n-2} \dots v_0)_b < (10\dots0) = b^{n-1}. \quad \square$$

Upper bound on $\hat{q}$

Lemma

1. If $v = b^{n-1} = (10\ldots0)_b$, then $q = \hat{q}$.
2. If $v \neq b^{n-1}$ then $\hat{q} < \frac{u}{v-b^{n-1}}$.

Theorem B

If $v_{n-1} \geq \lfloor b/2 \rfloor$, then $\hat{q} \leq q + 2$.

Proof. Assume $\hat{q} \geq q + 3$. By Lemma (1), $v - b^{n-1} \neq 0$, and

$$3 \leq \hat{q} - q < \frac{u}{v - b^{n-1}} - \left(\frac{u}{v} - 1\right) = \frac{uv - uv + ub^{n-1}}{v(v - b^{n-1})} + 1 = \frac{u}{v}\left(\frac{b^{n-1}}{v - b^{n-1}}\right) + 1.$$

Therefore

$$\frac{u}{v} > 2\left(\frac{v - b^{n-1}}{b^{n-1}}\right) \geq 2\left(\frac{v_{n-1}b^{n-1} - b^{n-1}}{b^{n-1}}\right) = 2(v_{n-1} - 1).$$

This implies $q = \lfloor u/v \rfloor \geq 2(v_{n-1} - 1)$.

Upper bound on $\hat{q}$ (cont.)

Theorem B

If $v_{n-1} \geq \lfloor b/2 \rfloor$, then $\hat{q} \leq q + 2$.

The inequalities $\hat{q} \leq b - 1$, $\hat{q} \geq q + 3$ and $q \geq 2(v_{n-1} - 1)$ imply

$$b - 4 \geq \hat{q} - 3 \geq q \geq 2(v_{n-1} - 1).$$

We have therefore

$$v_{n-1} \leq b/2 - 1 < \lfloor b/2 \rfloor,$$

which finishes the proof. □

Corollary

If $v_{n-1} \geq \lfloor b/2 \rfloor$, then $\hat{q} - 2 \leq q \leq \hat{q}$.

- Important: The conclusion is **independent** on the radix b.
- The condition that $v_{n-1} \geq \lfloor b/2 \rfloor$ is a **normalization requirement.**
- To ensure it, **multiply both** u **and** v by $d = 2^k$ such that $\lfloor b/2 \rfloor b^{n-1} \leq dv < b^n$.

Algorithm C to find q

Algorithm C (Division of an $n+1$-digit number by an n-digit number)

Given positive integers u, v in base b:

$$u = (u_n u_{n-1} \ldots u_0)_b$$
$$v = (v_{n-1} v_{n-2} \ldots v_0)_b$$

We assume $u/v < b$ and $v_{n-1} \geq \lfloor b/2 \rfloor$. The following algorithm returns $\lfloor u/v \rfloor$.

1. Set $q \leftarrow \min(\lfloor (u_n b + u_{n-1})/v_{n-1} \rfloor, b-1) - 2$ and $r \leftarrow u - qv$.
2. While $r \geq v$, set $q \leftarrow q + 1$ and $r \leftarrow r - v$.
3. Return q.

Important: By the Corollary, no *matter how large the radix b is,* one makes **step (2) at most twice.**

Remark: One can make Algorithm C more efficient by testing on v_{n-2}.

Outline

Algorithm D – division of non-negative integers

Given non-negative integers

$$u = (u_{m+n-1} u_{m+n-2} \dots u_1 u_0)_b, \qquad v = (v_{n-1} v_{n-2} \dots v_1 v_0)_b,$$

where $v_{n-1} \neq 0$ and $n > 1$. We form the radix-b quotient $q = \lfloor u/v \rfloor$ and the remainder $r = u \pmod v$

$$q = (q_m q_{m-1} \dots q_1 q_0)_b, \qquad r = (r_{n-1} r_{n-2} \dots r_1 r_0)_b.$$

Algorithm D

1. (Normalize.) Set $d \leftarrow 2^k$ such that $\lfloor b/2 \rfloor b^{n-1} \leq dv < b^n$. Set $(u_{m+n} u_{m+n-1} \dots u_1 u_0)_b \leftarrow du$. Set $(v_{n-1} v_{n-2} \dots v_1 v_0)_b \leftarrow dv$.
2. (Initialize j.) Set $j \leftarrow m$.
3. (Calculate q_j.) Set the subarray $u^* \leftarrow (u_{j+n} u_{j+n-1} \dots u_j)_b$. Apply Algorithm C with input u^* and v to compute q_j.
4. (Multiply and subtract.) Replace the subarray u^* by $u^* - q_j v$.
5. (Loop on j.) Decrease j by one. Now if $j \geq 0$, go back to (D3).
6. (Unnormalize.) Return q as quotient and u^*/d as remainder.