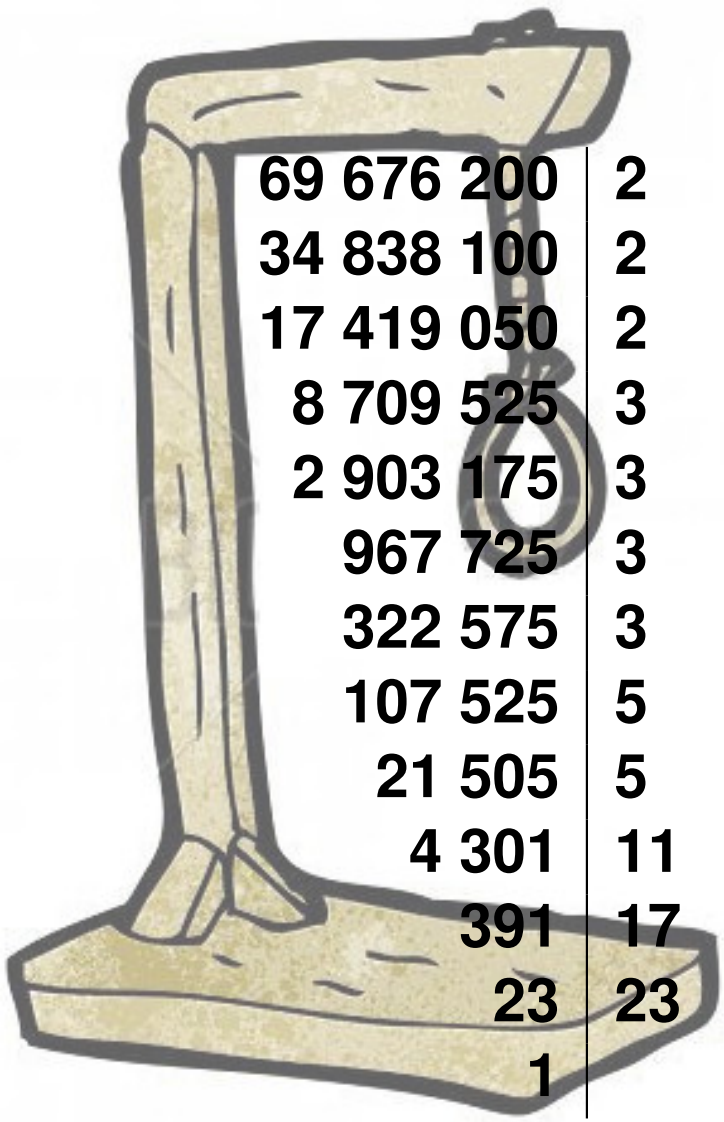
Egy veszélyesen sokat érő rejtvény

Prímszám-e az alábbi N?
Mi N prímtényezős felbontása?

$N = 24666595264313974103506253388560710289239$
$60513289163096920589859699784796360131211104 4$
$37606750062063154678807239444780455924743968 5$
$86217893880368234241976751703064371087567967 8$
$35364453968847839133363971093193672132166237 0$
$74766362818993796221219195282601966259462080 9$
$02665616981653955141760978590281574983756675 4$
$64503848327747118876276705669294327550746012 4$
$31011109726313365788217228346649662443560190 0$
$88110558950488160099117705885146509428075259 5$
$75652092499917140169223940161763450286380190 4$
$44478746509006228772739582543573943051774785 5$
$97040735374251406122314742129628144597826490 7$
$24428165803074386418398897234152778 3$

Akasztófa módszer



$$N = a \cdot b, \quad a \leq b$$
$$\Downarrow$$
$$a \leq \sqrt{N}$$

Ha $N \approx 2,47 \times 10^{616}$, akkor

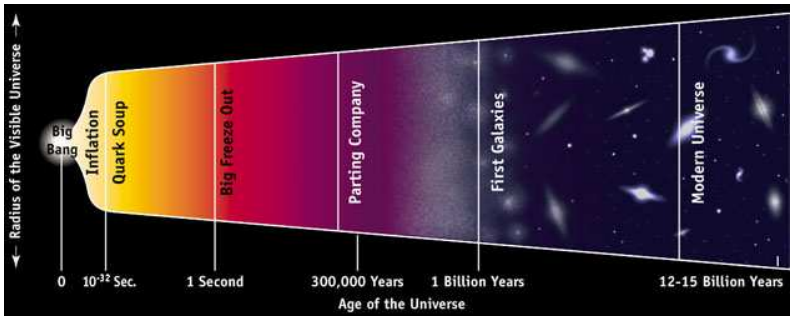
$$\sqrt{N} \approx 1,57 \times 10^{308}$$

osztásra is szükség lehet (ha N prím vagy két nagyon közeli prím szorzata).

Katasztrófa módszer

- $N \approx 2,47 \times 10^{616} \Rightarrow \sqrt{N} \approx 1,57 \times 10^{308}$ osztás
- A leggyorsabb sci-fi szuperszámítógép:
  - az információ fénysebességgel terjed ($3 \times 10^8 m/s$)
  - egy processzor mérete: egy hidrogén atom átmérője ($5 \times 10^{-11} m$)
  - processzorok száma: a protonok száma az Univerzumban ($1,575 \times 10^{79}$)
  - $\Rightarrow 9,45 \times 10^{97}$ művelet ($\approx$ osztás) másodpercenként
  - Az $1,57 \times 10^{308}$ osztás időigénye: $\approx 6,02 \times 10^{211}$ másodperc

- Az Univerzum életkora: 4.33×10^{17} másodperc



Mit nevezzünk hatékony algoritmusnak?

A definíció...

- ...nem függhet egy konkrét számítógéptől;
- ...nem adhat meg univerzális „időkorlátot”;
- ...meg kell engedje, hogy a futásidő függjön az input méretétől.

Az input mérete: $\approx$ a tárolásához szükséges memória mérete (pl. bájtok száma).

$N = 24666595264313974103506253388 5...$ (617 számjegy)

Az input mérete: $n = 617$

Mit nevezzünk hatékony algoritmusnak?

Az input mérete: n	„A” Algoritmus futásideje: 1.2^n	„B” Algoritmus futásideje: $100 \cdot n^5$
20	38,34	$3,2 \times 10^8$
50	$9,1 \times 10^3$	$3,13 \times 10^{10}$
100	$8,28 \times 10^7$	1×10^{12}
200	$6,86 \times 10^{15}$	$3,2 \times 10^{13}$
1000	$1,52 \times 10^{79}$	1×10^{17}
2048	$1,46 \times 10^{162}$	$3,6 \times 10^{18}$

Egy algoritmust általában akkor tekintünk hatékonynak, ha a lépésszáma ($\approx$ futásideje) felülről becsülhető az input méretének egy ***polinomjával***:

lépésszám $\leq c \cdot n^k$ valamilyen c-re és k-ra.

Az ilyen algoritmusokat ***polinomiális futásidejűnek*** nevezzük.

Mit nevezzünk hatékony algoritmusnak?

Az input mérete: n	„A” Algoritmus futásideje: 1.2^n	„B” Algoritmus futásideje: $100 \cdot n^5$
20	38,34	$3,2 \times 10^8$
50	$9,1 \times 10^3$	$3,13 \times 10^{10}$
100	$8,28 \times 10^7$	1×10^{12}
200	$6,86 \times 10^{15}$	$3,2 \times 10^{13}$
1000	$1,52 \times 10^{79}$	1×10^{17}
2048	$1,46 \times 10^{162}$	$3,6 \times 10^{18}$

Egy algoritmus exponenciális futásidejű, ha minden n-re van olyan n méretű input, amin a lépésszáma $\geq a^n$, ahol $a > 1$ fix.

Az exponenciális futásidejű algoritmusok **NEM** polinomiális futásidejűek!

(Ugyanis: minden rögzített $a > 1$ és $c, k \geq 0$ esetén van olyan n_0 küszöb, hogy $a^n > c \cdot n^k$, ha $n \geq n_0$. $\longrightarrow$*Analízis*)

Vannak más, **nem** polinomiális futásidejű algoritmusok is, az exponenciális futásidő erre csak egy (gyakori) példa.

Számelméleti algoritmusok

Számelméleti algoritmus: az input és az output is néhány egész számból áll.

Példa: írásbeli összeadás (Input: $a, b \in \mathbb{N}$, Output: $a + b$)

$$\begin{array}{r} 376067500620631546788072394447804559247439685 \\ + \quad 178938803682342419767517030643710875679678 \\ \hline \ldots 70123119363 \end{array}$$

a: k jegyű, b: ℓ jegyű (és $k \geq \ell$) $\Longrightarrow$ input mérete: $n = k + \ell$

Az írásbeli összeadás jegyenként fix számú, mondjuk legföljebb c darab elemi bitműveletet végez.

Ezért a lépésszáma $\leq c \cdot k \leq c \cdot n$

Így az írásbeli összeadás polinomiális futásidejű algoritmus.

(Sőt: ***lineáris futásidejű***: a lépésszáma $\leq c \cdot n^1$.)

Számelméleti algoritmusok

Példa: akasztófa módszer (Input: $N \in \mathbb{N}$)

N: n jegyű $\implies$ $10^{n-1} \leq N < 10^n$

$\sqrt{N}$ osztásra is szükség lehet:

$$\sqrt{N} \geq \sqrt{10^{n-1}} = \left(\sqrt{10}\right)^{n-1} > 2^n \quad \text{(ha } n \geq 3\text{)}$$

Ezért az akasztófa módszer exponenciális futásidejű és így a gyakorlati alkalmazásokban használhatatlan (legalábbis „nagy” számokra).

Ha egy számelméleti algoritmus lépésszáma magukkal az input szám(ok)kal (vagy pl. azok négyzetgyökével) arányos, akkor az exponenciális futásidejű.

Ugyanis: az input méretétől, vagyis a számjegyek számától maga a szám exponenciálisan függ.

Számelméleti algoritmusok

Mennyi a számjegyek száma (vagyis az input mérete)?

a: n jegyű $\implies$ $10^{n-1} \leq a < 10^n$

$n - 1 \leq \lg a < n$

Az a számjegyeinek száma 10-es számrendszerben: $\approx \lg a$.

Ha a számrendszer alapja t (például $t = 2$ vagy $t = 16$, stb): az a számjegyeinek száma $\approx \log_t a$.

Például 10-esről 2-es számrendszerre áttérve:

$$\log_2 a = \frac{\log_{10} a}{\log_{10} 2} \approx 3,32 \cdot \log_{10} a.$$

Ezért abból a szempontból, hogy az algoritmus polinomiális futásidejű-e **teljesen mindegy**, hogy milyen számrendszerben van megadva az input: az input méretét a számrendszerváltás csak egy rögzített konstansszorosára változtatja.

Számelméleti algoritmusok

Összefoglalva:

A számelméleti algoritmusok esetében

- az input mérete az input számok összes számjegyeinek száma,
- ami lényegében azonos az input számok logaritmusainak összegével (ahol a logaritmus alapja most mindegy);
- az algoritmus akkor polinomiális futásidejű, ha a lépésszáma $\leq c \cdot n^k$, ahol n az input számok összes számjegyeinek száma (és c és k rögzített);
- ha az algoritmus lépésszáma például az egyik input számmal arányos, akkor az exponenciális futásidejű és így **NEM** polinomiális futásidejű, a gyakorlatban (kicsi inputoktól eltekintve) **nem hatékony**.