Treba li vjerovati računalu?
Računalo je izračunalo...

Sanja i Saša Singer

Računalo je izbacilo... Neće mi prihvatiti...

Koliko ste puta ove dvije rečenice čuli na TV-u, u banci ili negdje drugdje? U oba slučaja, računalo je personificirano kao nadnaravno sposobna osoba, koja je u prvom slučaju bezgrešna, a u drugom odbija nešto učiniti!

Treba li takvim tužbalicama vjerovati? I tko je krivac? Računalo ili ljudi koji su mu naredili da se upravo tako ponaša? Istina je da smo u 2001. godini, ali vaše računalo nije (svojeglavi) HAL 9000 iz "2001. odiseje u svemiru" (a i njemu su ljudi pomogli da postane ubojica).

Navedene dvije rečenice najčešće su samo jadno pokriće nečije nesposobnosti. Službeniku za šalterom u banci vjerojatno treba oprostiti, jer on samo izražava svoju bespomoćnost, a pravi krivac je negdje daleko.

Puno je opasnije kad čujete "Računalo je izbacilo..." od strane inženjera i znanstvenika kao glavno opravdanje budućih važnih projekata. Tad nas hvata strah! Zašto? Sama rečenica pokazuje da dobivene rezultate nitko nije pogledao, nego da im se slijepo vjeruje. A oni mogu biti pogrešni iz razno-raznih razloga, a najčešći krivac nije računalo.

Iz osobnog iskustva znamo da je provjera dobivenih rezultata najbolnija točka nastave numerike, u koju je slušače najteže uvjeriti. Metode je manje-više lako naučiti. U praksi, brojevi uvijek imaju neko značenje, izvora grešaka je puno – javljaju se na svakom koraku, a analiza grešaka u rezultatima katkad je vrlo sofisticirana. Slijepo vjerovanje rezultatima može biti pogibeljno.

Izvori grešaka su:

- model,
- metoda za rješavanje modela,
- ulazni podaci (mjerenja),
- aritmetika računala.

Sve četiri vrste grešaka lako je razumjeti. Međutim, za posljednju, vrlo je teško vjerovati da ona može biti toliko značajna — dominantna u odnosu na ostale, tako da je rezultat zbog nje besmislen.

No, ipak treba priznati i računala, iznimno rijetko, ali griješe. Koliko je nama poznato, u novija vremena poznata je samo greška dijeljenja u jednoj seriji Pentium procesora (1994. godine).

Aritmetika računala

U računalu postoje dva bitno različita tipa brojeva: cijeli brojevi i realni brojevi. Oba skupa su **konačni podskupovi** odgovarajućih skupova $\mathbb{Z}$ i $\mathbb{R}$ u matematici. Kao baza za prikaz oba tipa koristi se baza 2.

Cijeli se brojevi prikazaju korištenjem n bitova – binarnih znamenki, od kojih jedna služi za predznak, a ostalih $n-1$ za znamenke broja. Matematički gledano, aritmetika cijelih brojeva u računalu je modularna aritmetika u prstenu ostataka modulo 2^n, samo je sistem ostataka simetričan oko 0, tj.

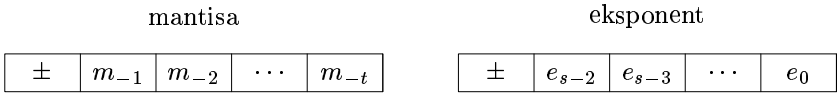
$$-2^{n-1}, \ldots, -1, 0, 1, \ldots, 2^{n-1} - 1.$$

Brojeve izvan tog raspona uopće ne možemo spremiti u računalo.

Realni brojevi prikazuju se korištenjem mantise m i eksponenta e:

$$r = \pm m \cdot 2^e,$$

pri čemu je e cijeli broj u određenom rasponu, a m racionalni broj za koji je $1/2 \leq m < 1$ (tj. mantisa započinje s $0.1\ldots$). Za $r = 0$, mantisa je 0. U računalu se eksponent prikazuje kao s-bitni cijeli broj, a za mantisu pamti se prvih t znamenki iza binarne točke.



Dakle, skup svih realnih brojeva prikazivih u računalu je omeđen, a možemo ga parametrizirati duljinom mantise i eksponenta i označiti s $\mathbb{R}(s,t)$.

Primijetite da se ne može svaki realni broj egzaktno spremiti u računalo. Pretpostavimo da je broj $x \in \mathbb{R}$ unutar prikazivog raspona i

$$x = \pm\Big(\sum_{k=1}^{\infty} b_{-k} 2^{-k}\Big) 2^e.$$

Ako mantisa broja ima više od t znamenki, bit će spremljena aproksimacija tog broja $fl(x) \in \mathbb{R}(t,s)$ koja se može prikazati kao

$$fl(x) = \pm\Big(\sum_{k=1}^{t} b^*_{-k} 2^{-k}\Big) 2^{e^*}.$$

Slično kao kod decimalne aritmetike (kad je prva odbačena znamenka ≤ 4 zaokružujemo nadolje, inače nagore), ako je prva odbačena znamenka 1, broj zaokružujemo nagore, a ako je 0 nadolje. Time smo napravili grešku manju ili jednaku 2^{-t-1+e}. Gledajući relativno, greška je manja ili jednaka

$$\left|\frac{x - fl(x)}{x}\right| \leq \frac{2^{-t-1+e}}{2^{-1} \cdot 2^e} = 2^{-t},$$

tj. imamo vrlo malu relativnu grešku. Veličinu 2^{-t} zovemo jedinična greška zaokruživanja (engl. unit roundoff) i uobičajeno označavamo s u.

Da bismo stekli osjećaj o veličinama o kojim govorimo, napišimo s i t i veličine koje se iz njih izvode za standardne tipove realnih brojeva:

	`single`	`double`	`extended`
duljina	32 bita	64 bita	80 bitova
duljina mantise	$23+1$ bit	$52+1$ bit	64 bita
duljina eksponenta	8 bitova	11 bitova	15 bitova
jedinična greška zaokruživanja	2^{-24}	2^{-53}	2^{-64}
	$5.96 \cdot 10^{-8}$	$1.11 \cdot 10^{-16}$	$5.42 \cdot 10^{-20}$
raspon	$10^{\pm 38}$	$10^{\pm 308}$	$10^{\pm 4932}$

Za sva tri tipa rezerviran je još jedan bit za predznak. Kod tipova `single` i `double` dodatni bit u duljini mantise je tzv. "sakriveni bit" (engl. hidden bit) jer je prvi znak iza binarne točke uvijek 1, pa se ne mora pamtiti. To je dogovoreni IEEE standard u kojem je propisano, ne samo kako se brojevi prikazuju u računalu, nego i svojstva aritmetike. Budimo pošteni, standard je nešto složeniji nego što smo ovdje opisali, ali ti dodatni detalji (kao što su recimo "zaštitne znamenke" (engl. guard digits)) nepotrebno zamagljuju bitno.

Osnovna pretpostavka je da za osnovne aritmetičke operacije ($\circ$ označava $+$, $-$, $*$, $/$) nad $x, y \in \mathbb{R}(s,t)$ vrijedi

$$fl(x \circ y) = (1+\varepsilon)\,(x \circ y), \quad |\varepsilon| \leq u, \tag{1}$$

za sve $x, y \in \mathbb{R}(s,t)$ za koje je $x \circ y$ u dozvoljenom rasponu. U protivnom, postoje rezervirani eksponenti koji označavaju "posebno stanje" (overflow, underflow, dijeljenje s 0 i nedozvoljenu operaciju kao što su $0/0$, $\sqrt{-1}$).

Oznaka $fl(\ )$ sad ima značenje rezultata dobivenog računalom za operaciju $x \circ y$. Model kaže da je izračunata vrijednost $x \circ y$ "jednako dobra" kao zaokružen egzaktni rezultat, u smislu da je u oba slučaja jednaka ocjena relativne greške. Model, međutim, ne zahtijeva da je za $x \circ y \in \mathbb{R}(s,t)$ greška $\varepsilon = 0$.

Desnu stranu relacije (1) možemo interpretirati i kao egzaktno izvedenu operaciju $\circ$ na malo perturbiranim podacima. Koje su operacije opasne ako nam je aritmetika egzaktna, a podaci malo perturbirani, tj. ako je $|\varepsilon_x|, |\varepsilon_y| \leq u$? Ako je $\circ$ množenje, imamo

$$x\,(1+\varepsilon_x) * y\,(1+\varepsilon_y) \approx xy\,(1+\varepsilon_x+\varepsilon_y) := xy(1+\varepsilon_*), \quad |\varepsilon_*| \leq 2u.$$

Ako imamo dijeljenje, vrijedi slično

$$\frac{x\,(1+\varepsilon_x)}{y\,(1+\varepsilon_y)} \approx \frac{x}{y}\,(1+\varepsilon_x)\,(1-\varepsilon_y) := \frac{x}{y}\,(1+\varepsilon_/), \quad |\varepsilon_/| \leq 2u.$$

Neka su x i y proizvoljnog predznaka. Za zbrajanje (oduzimanje) vrijedi:

$$x\,(1+\varepsilon_x) + y\,(1+\varepsilon_y) = x + y + x\varepsilon_x + y\varepsilon_y := (x+y)\Big(1 + \frac{x\varepsilon_x + y\varepsilon_y}{x+y}\Big),$$

uz pretpostavku da $x + y \neq 0$. Definiramo

$$\varepsilon_{\pm} := \frac{x\varepsilon_x + y\varepsilon_y}{x+y} = \frac{x}{x+y}\,\varepsilon_x + \frac{y}{x+y}\,\varepsilon_y.$$

Ako su x i y brojevi istog predznaka, onda je

$$\left|\frac{x}{x+y}\right|, \left|\frac{y}{x+y}\right| \leq 1, \tag{2}$$

pa je $|\varepsilon_{\pm}| \leq 2u$. U suprotnom, ako x i y imaju različite predznake, kvocijenti u (2) mogu biti proizvoljno veliki kad je $|x+y| \ll |x|, |y|$.

Možemo zaključiti da **opasnost** nastupa ako je rezultat zbrajanja brojeva suprotnog predznaka broj koji je po apsolutnoj vrijednosti mnogo manji od polaznih podataka. Dakle, čak i kad bi aritmetika računala bila egzaktna, zbog početnog zaokruživanja, rezultat može biti (i najčešće je) pogrešan.

Pokažimo to na jednostavnom primjeru računala u bazi 10. Pretpostavimo da je mantisa duga 4 dekadske znamenke, a eksponent dvije. Neka je

$$x = 0.88866 = 0.88866 \cdot 10^0, \quad y = 0.88844 = 0.88844 \cdot 10^0.$$

Umjesto brojeva x i y, spremili smo najbliže prikazive, tj.

$$fl(x) = 0.8887 \cdot 10^0, \quad fl(y) = 0.8884 \cdot 10^0$$

i napravili malu relativnu grešku. Budući da su im eksponenti jednaki, možemo oduzeti znamenku po znamenku mantise, a zatim normalizirati i dobiti

$$fl(x) - fl(y) = 0.8887 \cdot 10^0 - 0.8884 \cdot 10^0 = 0.0003 \cdot 10^0 = 0.3??? \cdot 10^{-3},$$

pri čemu upitnici predstavljaju znamenke koje više ne možemo restaurirati, pa računalo na ta mjesta upisuje 0. Primijetimo da je pravi rezultat $0.22 \cdot 10^{-3}$, pa je već prva značajna znamenka pogrešna, a relativna greška velika!

Iako je sama operacija oduzimanja bila egzaktna za $fl(x)$ i $fl(y)$, rezultat je pogrešan. Na prvi pogled čini nam se da znamo bar red veličine rezultata i da to nije tako strašno. Prava katastrofa nastupa ako $0.3??? \cdot 10^{-3}$ uđe u naredna zbrajanja i oduzimanja i ako se pritom "skrati" i ta trojka. Tada nemamo nikakve kontrole nad rezultatom.

Školski primjeri

Primjer 1. Vrijedost $y = \sqrt{x+\delta} - \sqrt{x}$ uz $|\delta| \ll x$, $x > 0$, nikad se ne računa po napisanoj formuli. Uvijek se pribjegava deracionalizaciji, tj.

$$y = (\sqrt{x+\delta} - \sqrt{x}) \cdot \frac{\sqrt{x+\delta} + \sqrt{x}}{\sqrt{x+\delta} + \sqrt{x}} = \frac{\delta}{\sqrt{x+\delta} + \sqrt{x}}.$$

Uočite da je opasno oduzimanje zamijenjeno benignim dijeljenjem i zbrajanjem. □

Primjer 2. Vrijednost $f(x) = (x-n)^{10}$, gdje je $n \in \mathbb{N}_0$ računamo računalom u okolini točke n, prvo direktno po formuli kako je napisano, a zatim korištenjem razvoja u Taylorov red oko točke 0. Što je točnije?

Formula potenciranja binoma (Taylorov red oko 0) je

$$(x-n)^{10} = \sum_{k=0}^{10} \binom{10}{k} x^k (-n)^{10-k}. \tag{3}$$

Prvo, primijetimo da je u okolini točke n lijeva strana (3) mali broj. Koeficijenti s desne strane (3) su alternirajući, što nas upozorava na oprez. Ako su još i veliki, možemo očekivati pogrešan rezultat. Binomni koeficijenti imaju maksimum u srednjem članu. Za x blizu n, po apsolutnoj vrijednosti najveći član desne strane u (3) približno je jednak $252n^{10}$, što raste s porastom n. U sumi mora doći do kraćenja, pa je rezultat bolje izračunati direktno. □

Primjer 3. Funkcije e^x i $\sin x$ imaju Taylorove redove oko točke 0 koji konvergiraju za proizvoljan $x \in \mathbb{R}$. Zbrajanjem dovoljno mnogo članova tih redova, možemo, barem u principu, dobro aproksimirati vrijednosti funkcije e^x i $\sin x$.

Ako to napravimo računalom, rezultat će biti zanimljiv. Greška metode (tj. greška odbacivanja) lako se računa. Za dovoljno glatku funkciju f, Taylorov red možemo aproksimirati Taylorovim polinomom

$$p(x) = \sum_{k=0}^{n} \frac{f^{(k)}(0)}{k!} x^k + R_{n+1}(x), \qquad R_{n+1}(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} x^{n+1},$$

pri čemu je ξ neki broj između 0 i x. Traženi Taylorovi polinomi su

$$e^x \approx \sum_{k=0}^{n} \frac{x^k}{k!}, \qquad \sin x \approx \sum_{k=0}^{n} \frac{(-1)^k x^{2k+1}}{(2k+1)!}.$$

Vrijedi

$$(e^x)^{(n)} = e^x, \qquad (\sin x)^{(n)} = \sin\left(x + \frac{n\pi}{2}\right),$$

pa su pripadne greške odbacivanja

$$R_{n+1}(x) = \frac{e^\xi x^{n+1}}{(n+1)!}, \qquad R_{2n+3}(x) = \frac{\sin(\xi + \frac{2n+3}{2}\pi) x^{2n+3}}{(2n+3)!},$$

Radi jednostavnosti, pretpostavimo da je $x > 0$. Iz $\xi \le x$ dobivamo ocjene

$$|R_{n+1}(x)| \le \frac{e^x x^{n+1}}{(n+1)!}, \qquad |R_{2n+3}(x)| = \left|\frac{\sin(\xi + \frac{2n+3}{2}\pi) x^{2n+3}}{(2n+3)!}\right| \le \left|\frac{x^{2n+3}}{(2n+3)!}\right|.$$

Zbrojimo li članove reda sve dok prvi odbačeni član ne padne ispod zadane točnosti ε, napravili smo grešku odbacivanja manju ili jednaku

$$\begin{cases} e^x\varepsilon & \text{za } e^x, \\ \varepsilon & \text{za } \sin x. \end{cases} \tag{4}$$

Izračunajmo $\sin(15\pi)$, $e^{15\pi}$, $\sin(25\pi)$ i $e^{25\pi}$ korištenjem Taylorovog reda oko 0 u `extended` preciznosti. Primjer je izabran tako da je $\sin(15\pi) = \sin(25\pi) = 0$, dok su druga dva broja vrlo velika. Prema (4), greška metode za računanje je u slučaju funkcije e^x relativno mala, a u slučaju funkcije $\sin x$, mala po apsolutnoj vrijednosti.

Izaberemo li $\varepsilon = 10^{-17}$, dobivamo (napisano samo prvih par znamenki rezultata)

$$\begin{aligned}
\sin(15\pi)_{funkcija} &= 9.3241 \cdot 10^{-18} & \exp(15\pi)_{funkcija} &= 2.9218 \cdot 10^{20} \\
\sin(15\pi)_{Taylor} &= -2.8980 \cdot 10^{-1} & \exp(15\pi)_{Taylor} &= 2.9218 \cdot 10^{20} \\
|\textit{greška odbacivanja}| &\leq 2.7310 \cdot 10^{-19} & |\textit{greška odbacivanja}| &\leq 2.7600 \cdot 10^{2} \\
\textit{relativna greška} &= 3.1081 \cdot 10^{16} & \textit{relativna greška} &= 1.4238 \cdot 10^{-18} \\
|\textit{maksimalni član}| &= 1.6969 \cdot 10^{19} & |\textit{maksimalni član}| &= 1.6969 \cdot 10^{19}
\end{aligned}$$

$$\begin{aligned}
\sin(25\pi)_{funkcija} &= 1.6697 \cdot 10^{-17} & \exp(25\pi)_{funkcija} &= 1.2865 \cdot 10^{34} \\
\sin(25\pi)_{Taylor} &= 3.0613 \cdot 10^{13} & \exp(25\pi)_{Taylor} &= 1.2865 \cdot 10^{34} \\
|\textit{greška odbacivanja}| &\leq 5.8309 \cdot 10^{-19} & |\textit{greška odbacivanja}| &\leq 2.3782 \cdot 10^{16} \\
\textit{relativna greška} &= 1.8334 \cdot 10^{30} & \textit{relativna greška} &= 7.0013 \cdot 10^{-19} \\
|\textit{maksimalni član}| &= 5.7605 \cdot 10^{32} & |\textit{maksimalni član}| &= 5.7943 \cdot 10^{32}.
\end{aligned}$$

Objašnjenje pojave je očito. Pogledamo li po apsolutnoj vrijednosti najveće brojeve koji se javljaju u računu, ustanovljavamo da su oni golemi. Za $\sin x$, rezultat je malen broj koji je dobiven oduzimanjem velikih brojeva, pa je netočan. Nasuprot tome, kod funkcije e^x, uvijek imamo zbrajanje brojeva istog predznaka, pa je rezultat točan. □

Primjer 4. Niti poredak operacija ne mora biti beznačajan. Zadan je linearni sustav

$$\begin{aligned} 0.0001\, x_1 + x_2 &= 1 \\ x_1 + x_2 &= 2. \end{aligned}$$

Matrica sustava je regularna, pa postoji jedinstveno rješenje $x_1 = 1.\dot{0}00\dot{1}$, $x_2 = 0.\dot{9}99\dot{8}$. Rješavamo li taj sustav računalom koje ima 4 decimalne znamenke mantise i 2 znamenke eksponenta, onda njegovo rješenje ovisi o poretku jednadžbi. Sustav zapisan u takvom računalu pamti se kao

$$\begin{aligned} 0.1000 \cdot 10^{-3}\, x_1 + 0.1000 \cdot 10^{1}\, x_2 &= 0.1000 \cdot 10^{1} \\ 0.1000 \cdot 10^{1}\ \ x_1 + 0.1000 \cdot 10^{1}\, x_2 &= 0.2000 \cdot 10^{1}. \end{aligned} \tag{5}$$

Prvo, riješimo sustav (5) Gaussovim eliminacijama. Množenjem prve jednadžbe s 10^4 i oduzimanjem od druge, dobivamo drugu jednadžbu oblika:

$$(0.1000 \cdot 10^1 - 0.1000 \cdot 10^5)\, x_2 = 0.2000 \cdot 10^1 - 0.1000 \cdot 10^5. \tag{6}$$

Da bi računalo moglo oduzeti odgovarajuće brojeve, manji eksponent mora postati jednak većem, a mantisa se denormalizira. Dobivamo

$$0.1000 \cdot 10^1 = 0.0100 \cdot 10^2 = 0.0010 \cdot 10^3 = 0.0001 \cdot 10^4 = 0.0000|1 \cdot 10^5,$$

ali za zadnju jedinicu nema mjesta u mantisi, pa je mantisa postala 0. Slično je i s desnom stranom. Zbog toga jednadžba (6) postaje

$$-0.1000 \cdot 10^5\, x_2 = -0.1000 \cdot 10^5,$$

pa joj je rješenje $x_2 = 0.1000 \cdot 10^1$. Uvrštavanjem u prvu jednadžbu, dobivamo:

$$0.1000 \cdot 10^{-3}\, x_1 = -0.1000 \cdot 10^1 \cdot 0.1000 \cdot 10^1 + 0.1000 \cdot 10^1 = 0.0000,$$

pa je $x_1 = 0$, što nije niti približno točan rezultat.

Promijenimo li poredak jednadžbi u (5), dobivamo

$$\begin{aligned} 0.1000 \cdot 10^1 \quad x_1 + 0.1000 \cdot 10^1\, x_2 &= 0.2000 \cdot 10^1 \\ 0.1000 \cdot 10^{-3}\, x_1 + 0.1000 \cdot 10^1\, x_2 &= 0.1000 \cdot 10^1. \end{aligned} \tag{7}$$

Množenjem prve jednadžbe s 10^{-4} i oduzimanjem od druge, dobivamo drugu jednadžbu oblika

$$(0.1000 \cdot 10^1 - 0.1000 \cdot 10^{-3})\, x_2 = 0.1000 \cdot 10^1 - 0.2000 \cdot 10^{-3}, \tag{8}$$

pa se (8) svede na $0.1000 \cdot 10^1\, x_2 = 0.1000 \cdot 10^1$, tj. $x_2 = 0.1000 \cdot 10^1$. Uvrštavanjem u prvu jednadžbu u (7) dobivamo

$$0.1000 \cdot 10^1\, x_1 = 0.2000 \cdot 10^1 - 0.1000 \cdot 10^1 \cdot 0.1000 \cdot 10^1 = 0.1000 \cdot 10^1,$$

pa je $x_1 = 0.1000 \cdot 10^1$, što točan rezultat korektno zaokružen na četiri decimalne znamenke. □

Primjeri iz prakse

Iako nam se čini nemogućim, ali numerika može biti opasna i izvan učionica. A onda to rezultira teškim ljudskim i materijalnim gubicima.

Promašaj raketa Patriot. U Zaljevskom ratu, 25. veljače 1991. Patriot rakete iznad Dhahrana u Saudijskoj Arabiji nisu uspjele pronaći i oboriti Scud projektil. Raketa Scud pala je na američku vojnu bazu usmrtivši 28 i ranivši stotinjak ljudi.

Izvještaj o katastrofi godinu dana poslije, rasvijetlio je okolnosti nesreće. Vrijeme se u računalu koje je upravljalo Patriot raketama brojilo desetinkama sekunde proteklim od trenutka od kad je računalo upaljeno. Kad desetinku prikažemo u binarnom prikazu, dobivamo

$$0.1_{10} = (0.0\dot{0}01\dot{1})_2.$$

Realne brojeve u tom računalu prikazivali su korištenjem mantise (nenormalizirane) duljine 24 bita. Spremanjem 0.1 u registar Patriot računala, napravljena je (decimalna) greška približno jednaka $9.5 \cdot 10^{-8}$.

Zbog stalne opasnosti od napada Scud raketama, računalo je bilo u pogonu 100 sati, što je $100 \cdot 60 \cdot 60 \cdot 10$ desetinki sekunde. Ukupna greška nastala greškom zaokruživanja je

$$100 \cdot 60 \cdot 60 \cdot 10 \cdot 9.5 \cdot 10^{-8} = 0.34\,\text{s}.$$

Ako je poznato da raketa Scud putuje brzinom 1676 m/s, onda su ga rakete Patriot pokušale naći više od pola kilometra daleko od njegovog stvarnog položaja. □

Eksplozija Ariane 5. Raketa Ariane 5 lansirana 4. lipnja 1995. iz Kouroua (Francuska Gvajana) nosila je u putanju oko Zemlje komunikacijske satelite (vrijednost 500 milijuna USD). 37 sekundi nakon lansiranja izvršila je samouništenje.

Dva tjedna kasnije, stručnjaci su objasnili događaj. Kontrolna varijabla (koja je služila samo informacije radi) u programu vođenja rakete mjerila je horizontalnu brzinu rakete. Greška je nastupila kad je program pokušao pretvoriti preveliki 64-bitni realni broj u 16-bitni cijeli broj. Računalo je javilo grešku, što je izazvalo samouništenje. Zanimljivo je da je taj isti program bio korišten u prijašnjoj sporijoj verziji Ariane 4, pa do katastrofe nije došlo. □

Potonuće naftne platforme Sleipner A. Naftna platforma Sleipner A potonula je prilikom sidrenja, 23. kolovoza 1991. u blizini Stavangera. Baza platforme su 24 betonske ćelije, od kojih su 4 produljene u šuplje stupove na kojima leži paluba. Prilikom uronjavanja baze došlo je do pucanja. Rušenje je izazvalo potres jačine 3.0 stupnja po Richterovoj ljestvici i štetu od 700 milijuna USD.

Greška je nastala u projektiranju, kad je upotrijebljena metoda konačnih elemenata s nedovoljnom točnošću (netko nije provjerio rezultate programa). Proračun je dao naprezanja 47% manja od stvarnih. Zanimljivo je da je nakon detaljne analize s točnijim konačnim elementima, izračunato da su ćelije morale popustiti na dubini od 62 metra. Stvarna dubina pucanja bila je 65 metara! □

Zaključak

Ponašanje rezultata se može predvidjeti — strah je nepotreban, a oprez nužan. Imamo li sumnju da je oduzimanjem brojeva došlo do kraćenja, svakako treba pokušati problem preformulirati ili provjeriti drugom metodom i/ili u višoj točnosti. Jasno je da treba poznavati i greške metode.