Knuth en Schuh

Donald Knuth is niet te stoppen. Men zou denken dat hij zo langzamerhand op zijn lauweren zou rusten, thuis zijn grote huiskamerorgel zou bespelen en slechts de deur uit zou gaan om nog ergens een eredoctoraat op te halen, maar niets is minder waar.

Afgelopen jaar woonde hij in Tsjechië de première bij van zijn multimediale orgelwerk Fantasia Apocalyptica, gebaseerd op het het laatste boek van het Nieuwe Testament *De openbaring van Johannes* en voorzien van illustraties door Duane Bibby. Ik vermoed dat het werk bij de doorsnee liefhebber van klassieke muziek even veel (of even weinig) weerklank zal vinden als Frank Zappa's uitstapjes naar het componeren van klassieke muziek (*The Yellow Shark*) maar het past wel in het idee dat computers programmeren een kunstvorm is als vele andere en zo'n uitvoering van Knuth's muziek is een geweldig mooie aanleiding voor gelijkgestemden om samen te komen en te genieten van elkaars gezelschap en goede luim.

Bovendien schrijft Knuth op volle kracht verder aan *The Art of Computer Programming*, een boek in vele delen als een gestaag uitdijend zonnestelsel, bestaande uit zijn denkwijze en leerstof. Het is een ‚work in progress' waarvan de voltooide delen ingebonden beschikbaar zijn terwijl kleinere deeltjes die vergevorderd zijn maar nog niet 'af' in paperback verschijnen, 'fascicle' genaamd en voorafgaande aan hun fysieke verschijning als voor-deeltje (pre-fascicle) in PostScript zijn te bekijken.

Een van die kleinste deeltjes in het Knuth universum is *The Art of Computer Programming, Volume 4, Pre-Fascicle 9B, A Potpourri of Puzzles* dat begin mei 2019 zichtbaar werd. Deel 4 gaat over ‚Combinatorial Algorithms', een onderdeel van computerwetenschap waar één goed idee een programma een miljoen keer sneller kan maken, aldus Knuth. De voorpublicatie van het 'puzzelboekje' is bedoeld als speels voorbeeld van de manier waarop deze aanpak in de praktijk kan werken.

Knuth opent zijn voorwoord met fraai geformuleerde bescheidenheid in klassieke traditie van welsprekendheid door de zeggen dat het niet zozeer zijn bedoeling is de lezer te imponeren met dit werk maar integendeel een kleine kring van lezers de mogelijkheid te bieden om fouten in zijn manuscript aan te wijzen voordat al te veel anderen deze zien ‚in de hoop dat ik mij, op een zekere dag, niet langer hoef te verontschuldigen voor wat nu nog slechts een hoopje schetsen is.'

7.2.2.8 A POTPOURRI OF PUZZLES 1

7.2.2.8. A potpourri of puzzles. Blah blah de blah blah blah. We'll discuss some of the most interesting time-wasters that have captured the attention of computer programmers over the years ... The "obvious" ways to solve them can often be greatly improved by using what we've learned in previous sections ...

* * *

Who knows what I might eventually say here?

* * *

Perfect digital invariants. In 1923, the great puzzlist Henry E. Dudeney observed that

$$370 = 3^3 + 7^3 + 0^3 \qquad \text{and} \qquad 407 = 4^3 + 0^3 + 7^3,$$

and asked his readers to find a similar example that doesn't have a zero in its decimal representation. A month later he gave the solution, $153 = 1^3 + 5^3 + 3^3$ [*Strand* **65** (1923), 103, 208] — curiously saying nothing about the obvious answer $371 = 3^3 + 7^3 + 1^3$. These examples were rediscovered independently by several other people, and eventually extended to mth powers of the digits for $m > 3$, and to radix-6 numbers for $b \neq 10$. Max Rumney [*Recreational Math. Magazine* #12 (December 1962), 6–8] mentioned $8208 = 8^4 + 2^4 + 0^4 + 8^4$, $(491)_{11} = 794 = 4^4 + 9^4 + 1^4$, ..., and named such numbers *perfect digital invariants* of order m.

Let $\pi_m x$ be the sum of the mth powers of the decimal digits of x. With this notation, the number x is a perfect digital invariant of order m in radix 10 if and only if $\pi_m x = x$. In particular, every order $m > 0$ has at least two perfect digital invariants, since the numbers 0 and 1 always qualify. And it turns out that most orders have *more* than two (see exercise 34), because of more or less random coincidences. For example, when $m = 100$ there's a unique third solution,

$$x = 2656162961933010980367641671003297920787484348544147 \\ 7176693876286933204788451137448014798509429584 = \pi_{100} x, \quad (20)$$

discovered in 2009 by Joseph Myers.

How can such a humongous number be found in a reasonable time? In the first place, we can always write $x = (x_m \ldots x_1 x_0)_{10}$, because exercise 30 shows that every mth-order solution has at most $m+1$ digits. In the second place, we can see that π_m depends only on the multiset $M_m(x) = \{x_m, \ldots, x_1, x_0\}$ of x's digits, not on the actual order of those digits. All we have to do, therefore, is look at each multiset, and see if $M_m(x_m^m + \cdots + x_1^m + x_0^m) = \{x_m, \ldots, x_1, x_0\}$.

A multiset of $m+1$ digits is what Section 7.2.1.3 calls a "multicombination," also known as a *combination of the ten objects* $\{0, 1, \ldots, 9\}$ *taken* $m+1$ *at a time with repetitions allowed*. If we renumber the subscripts by sorting the digits into order, such a multicombination is nothing more nor less than a solution to

$$9 \geq x_m \geq \cdots \geq x_1 \geq x_0 \geq 0. \quad (21)$$

May 3, 2013

PDE: A perfect digital invariant
Perfect digital invariants
Dudeney
powers of the digits
radix-b numbers
Rumney
perfect digital invariants
narcissistic numbers
coincidences
Myers
multiset
multicombination
combination
sorting

The Art of Computer Programming, Volume 4, Pre-Fascicle 9B, A Potpourri of Puzzles, pagina 5

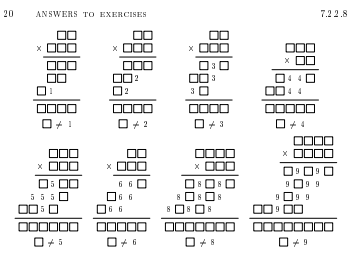
Zulk een bescheidenheid vergt een zuiver gevoel voor balans. Het moet oprecht gemeend zijn en geen vluchtig vermomde verwaandheid maar men moet ook weer niet te ver gaan in het afwaarderen van het eigen werk want waarom zou de lezer eraan beginnen als de auteur het zelf niks vindt? In de Nederlandse literatuur is de ontstaansgeschiedenis van P.C. Boutens' *Strofen uit de nalatenschap van Andries de Hoghe* een aangrijpend voorbeeld hiervan. Een jonge dichter, Jan S. van Drooge, zond Boutens wat werk ter beoordeling en schreef er overdreven bescheiden bij dat ‚de betere gedichten van zijn hand niet mede waren ingesloten, daar hij de tijd voor dat werk nog niet gekomen achtte.' Boutens, in een korzelige bui, zond alles per kerende post terug aangezien ‚het zinloos was verzen te lezen, aan welke de schrijver zelf al voorbij was.' Korte tijd later toen Boutens vernam dat de jonge man een einde aan zijn leven had gemaakt, voelde hij zich daaraan schuldig en jaren daarna eerde hij de jongeling door een verzenbundel min of meer aan deze toe te schrijven. (Zie ook Johan Polak, *Het voorbeeldige boek*, uitg. Balans 2006, blz 79, – overigens met behulp van ConTEXt getypeset door Willi Egger).

Knuth intussen is sympathiek, wars van arrogantie en hoffelijk naar zijn lezers: ‚het zou mij verbazen als het 'tagging algorithm' in antwoord 39 niet al eens ergens eerder is gepubliceerd, al denk ik niet dat ik het ooit eerder heb gezien.' Voorts biedt hij 32 dollarcent voor als iemand een waardevolle suggestie heeft voor tekstverbetering en zelfs 'onsterfelijke glorie' komt binnen bereik indien je naam daarna in het boek zal worden opgenomen.

Het boekje zelf is duizelingwekkend doordat je, als je de tekst leest, aanvankelijk de indruk kijgt dat het allemaal heel goed is te volgen en zelfs anekdotisch is, zoals het voorbeeld van 'mottige vermenigvuldiging' waarbij een veel van de cijfers onzichtbaar is. Het leest vlot weg tot je je realiseert dat je geen idee hebt wat er gebeurt. Er zijn kennelijk mensen die zo slim zijn dat zelfs onzichtbare getallen herkend worden.

In het hoofdstuk 'Discrete dissections' wordt iets besproken dat ik me vaag herinner uit mijn Montessori lagere schooltijd. Een uit kleurige vlakjes opgebouwd vierkant moet worden opgedeeld in twee kleinere vierkanten waarbij de keurpatronen gelijk blijven. Op school was het idee dat je door het maken van dergelijke 'taakjes' spelenderwijs leerde rekenen. Het rekenen heb ik er niet van geleerd maar het spelen des te beter. Knuth laat zien hoe je dergelijke vraagstukken op eenvoudige wijze wiskundig kan oplossen. Voor deze puzzles ontwierp Knuth een toepasselijk font, FONT36.

Knuth verwijst ook naar Fred Schuh (1875-1966), een Nederlandse wiskundige waar ik nog nooit van had gehoord. Online lees ik dat Schuh als kleurrijke radiopersoonlijkheid bekend stond om zijn voordrachten ‚Hoe leert men denken?' In Delft waren naar verluidt zijn mechanica-colleges van een niet te overtreffen helderheid en zijn meesterwerk *Wonderlijke problemen, Leerzaam tijdverdrijf door puzzle en spel* vormde voor Knuth wellicht de inspiratie voor juist deze fascicle.



20 ANSWERS TO EXERCISES 7.2.2.8

decimal arithmetic
bignum
stamping+
inverse permutation
deletion from lists

Fig. A–30. Skeleton multiplication puzzles with d ds.

(By the way, the answers to the given puzzles are 237457×720845 and $\mathrm{K} = 9$; 467224×6521 and $\mathrm{K} = 3$. Another nice puzzle with slack 0 is answered by 38522×3597003 and $\mathrm{K} = 6$. There are none with slack 0 and $z = 0$.)

55. Take the skeleton of 38522×3597, with $\mathrm{K} = 6$.

56. Instead of using the computer's built-in multiplication, it's best to implement decimal arithmetic from scratch. Say that a *bignum* is a nonnegative integer x that's represented as a sequence of bytes $x_0 x_1 \ldots x_{N-1}$, with $(\text{say})\ N \approx 25$; the value of x is $(x_{\ell} \ldots x_1 x_0)_{10}$, where $\ell = x_1$, and $x_{\ell} \neq 0$ unless $\ell = 0$. It's easy to write a routine that computes $x + 10^q y$, given bignums x and y and an offset q, and to prepare the basic multiplication table of bignum constants $a \cdot b$ for $0 \le a, b < 10$.

We maintain an array `BA[U][j]` of bignums, representing $j \cdot (a_l \ldots a_0)_{10}$ at level l of the algorithm, for $0 \le j < 10$. Clearly `BA[U][j]` $= \mathtt{BA}[U-1][j] + 10^l (j \cdot a_l)$ when $l > 0$. (See (56) and (57); but we don't truncate to l digits as shown there.) These values need to be computed only when j is a potentially useful multiplier digit. So we have another array `STAMP[U][j]` by which we can tell if `BA[U][j]` is valid (see below).

Next there's `CHOICE[k]`, for $0 \le k < m$, which is a permutation of $\{0, 1, \ldots, 9\}$; also `WHERE[k]`, which is the inverse permutation. (Thus `CHOICE[k][i]` $= j$ if and only if `WHERE[k][j]` $= i$.) The multiplier digits that haven't been ruled out by constraint p_k at level l are the first `S[l][k]` elements of `CHOICE[k]`, namely the elements j such that `WHERE[k][j]` $<$ `S[l][k]`. This setup permits easy deletion from lists while backtracking, because p_k becomes stronger as l increases; see 7.2.2–23.

Finally we prepare an array `ID` such that $p_k = p_{k'}$ if and only if `ID[k]` $=$ `ID[k']`. A `STACK` is used to propagate forced constraints. And the variable `HOUSES`, initially 0, holds ten times the serial number of the current node.

May 2, 2019

‚Knuth's keyboard' zoals door hem geschonken aan het Computer History Museum https://www.computerhistory.org/collections/catalog/102667297

Frans Goddijn