THE AUDIOACTIVE PACKAGE

R.A. LITHERLAND

1. Introduction

This is the documentation for programs in the the audioactive package. The purpose of these programs is explained in the introduction to my article [L], which should be read first.

2. The program `proof1`.

The Cosmological Theorem asserts that every string has finite longevity, and that there is an upper bound on their longevities. We call the maximum longevity of all strings the *cosmological constant*. (This term might better be used to describe the growth-rate constant λ of [C], but that is now universally known as Conway's Constant.) The program `proof1`, when run with suitable arguments, verifies that the cosmological constant is at most 24. That it is exactly 24 then follows from Guy's example given in [C]. The basic idea of this program is taken from Ekhad and Zeilberger's [EZ], though the details differ. (It also runs somewhat faster; with most reasonable arguments, around 2 minutes, as opposed to a week or two.) One difference is related to their statement that "Conway proved that it suffices to consider strings on $\{1,2,3\}$". In a sense, this is true. By day 2, any atom contains at most one digit greater than 3, as its last digit, and replacing it by 2 if the second-to-last digit is not 2, or 3 if it is, yields an atom that evolves in the same way. It follows easily that if all strings on $\{1,2,3\}$ have longevity at most N, then all strings whatsoever have longevity at most $N+2$. I have been unable to find any way to remove the "$+2$", however. Since there are strings on $\{1,2,3\}$ of longevity 24, I find myself forced to allow larger digits in the program. For $S = d_1^{e_1} d_2^{e_2} \dots d_n^{e_n}$, let $H(S)$ be the set of all d_i and e_i that are greater than 3. We say that strings S and T are *isotopic* if there is a bijection $H(S) \to H(T)$ that, when applied to both the digits and exponents in the above representation of S, transforms S to T. Then S_n is isotopic to T_n for all n, which means that it suffices to consider just one representative of each isotopy class.

One argument to `proof1` is required; it is a positive integer N. We compute succesively sets $A(N,1), A(N,2), \dots$ of strings, where $A(N,l)$ contains (up to isotopy) all atoms of length l that can appear as chunks of N-day old strings (and some that can't). We stop if some $A(N,l)$ is empty. For small N, this process will never terminate, but for large enough N it turns out that it does. (As in [EZ], $N = 9$ works.) The program will reject any values of N less than three, because this allows some simplifications. The process starts with $A(N,1) = \{1,2,3,4\}$. Note that a substring of an atom need not be an atom (2133 is an atom, but 213 is not), but a terminal substring must be. This means that having found $A(N,l)$, we can

Date: April 3, 2003.

construct $A(N, l+1)$ by considering all single-digit extensions xS with $1 \le x \le 3$ of each $S \in A(N, l)$, rejecting first those that split as $x.S$, and then (some of) those that cannot be chunks of N-day-old strings. (Here we do not need $x > 3$ because xS would split, or be less than two days old, in that case.) Rejecting split strings is easy; we just check the conditions of the Splitting Theorem. To describe the second test, we need some terminology. Recall that a (complete) 1-day-old string is parsed by inserting commas to show its descriptive nature, as in ,41,13, for example. Following [EZ], call a chunk of a 1-day-old string *female* if it starts just after a comma, and *male* if not. Also let $\max(S)$ be the maximum of 3 and all the digits of S. Suppose $n > 0$. A string S can be a male chunk of an n-day-old string iff some string xS can be a female chunk of one. If $n > 1$, this can only happen for $x \le 3$, while if $n = 1$ the value of x is irrelevant. Similarly, an odd-length string S can be a female chunk iff some Sx can; here it is enough to consider $1 \le x \le \max(S) + 1$. Now consider an even-length string $S = e_1 d_1 \dots e_k d_k$. For S to appear as a female chunk of an n-day-old string, it must have $d_i \ne d_{i+1}$ for $1 \le i < k$. If this is so, it describes the string $d_1^{e_1} \dots d_k^{e_k}$, and it is necessary that this string can appear as a chunk of an $(n-1)$-day-old string. (This is not a sufficient condition, but testing a sufficient condition is prohibitively slow.)

Assuming that N was chosen so that $A(N, L)$ is empty for some L, the cosmological theorem is true if all strings in $\bigcup_{1 \le l < L} A(N, l)$ have finite longevity, and an upper bound for the cosmological constant is the maximum longevity of these strings plus N. At this point, the second (optional) argument to `proof1` comes in; this specifies a level of stringency, which may be 1, 2 or 3 (with default 1). At each level, a certain class of interesting strings in each $A(N, l)$ is singled out, in such a way that taking the maximum longevity of only the interesting strings also gives an upper bound for the cosmological constant. For each l, the program prints out three things: the number of strings surviving at this length (the size of $A(N, l)$), the number of interesting strings found so far, and N plus the maximum longevity of those strings. (The last number is somewhat misleadingly labelled "cosmological constant"; its final value is an upper bound for this.) At level 1, we just check if the longevity is at least $24 - N$; this gives the same final bound as if we took the maximum over all strings in $\bigcup A(N, l)$. Running the program with $N = 9$ and level 1 gives a bound of 27.

At level 2, we further test if a string can appear as an atomic constituent of an N-day-old string. (Consider that 1 is an atom, and a chunk of 13, but is not a constituent of 13.) Note that the test for n-day-old chunks above is easily modified to test whether a string can be an initial, or final, chunk, or both. Given S, we can construct finite sets X_1, X_2, X_3 and X_4 of strings extending S, such that S can appear as a constituent of an N-day-old string iff some string from X_1 is a complete N-day-old string, or a string from X_2 is an initial chunk of one, or a string from X_3 is a final chunk of one, or a string from X_4 is an arbitrary chunk of one, giving necessary conditions for S to be a constituent. (The construction of these sets is fiddly, and the reader wanting the gory details is referred to the code and comments for the functions `can_split_before()` and `can_split_off()` in `proof1.c`. This is where the condition $N \ge 3$ comes in. In a 3-day-old string, a digit greater than 3 can only follow a 1, which allows us to cut down the number of extensions to be considered.) Running the program with $N = 9$ at level 2 produces the sharp bound 24.

Obviously, level 3 cannot improve the bound, but it does show up a couple of interesting phenomena. At this level, each string that is interesting at level 2 is subjected to a necessary and sufficient condition for being a constituent of an N-day-old string. To do this, we just need to observe that in order for an even-length string to appear as a female chunk of an n-day-old string, it is necessary and sufficient for the parental string to appear in an $(n-1)$-day-old string in which it is:

(1) either initial, or preceeded by a digit different from its first; and
(2) either terminal, or followed by a digit different from its last.

This can clearly be tested, but it is very slow, so it is best to check that level 2 for a given N produces a smallish number of interesting strings before trying level 3. There is a final optional argument, a switch `-p`. This causes the program to print a list of the interesting strings, and at level 3 an ancestor of each of them (which is guaranteed to have longevity 24). With $N = 11$ and level 3, the program produces 6 interesting strings, 3 of them being the cases $n = 2$, 3 and 4 of a family of the form $S1n$. Strings which yield these at day 11 are:

$$\begin{aligned} & 33322211n \quad (n > 1), \\ & 3332221131, \\ & 33322211311211, \\ \text{and} \quad & 33322211341111. \end{aligned}$$

Examining the evolution of these strings, one sees that at day 14, each has produced the atom 3123222, and that this is the only atom to survive to day 24. Indeed, running the program with $N = 14$ at level 3 confirms that this is the only string of longevity 10 that can appear as a constituent of a 14-day-old string. (This takes about 20 minutes.) One can produce more atomic strings of longevity 24 in a couple of ways. First, any string of the form $333222113nm^n$ with $n > 3$ and $m \neq n$ behaves essentially the same as the last string above. Second, preceding any of the above strings by a string S ending in either a 1 or 2^2 does not change the longevity, because the part of the string affected by S splits off at day 11.

3. The program `proof2`

As explained in the proof of Theorem 1 of [L], there is a certain set S of 3360 strings such that, if every string in S has longevity at most 10, then the cosmological constant is at most (hence exactly) 24. This program takes no arguments, and prints all the exotic (neither common nor transuranic) atoms that appear in the strings of S, together with their longevities and maximum longevity.

4. The program `evolve`

This program takes no arguments. When run, it waits for you to enter a digit string, and then shows its evolution into common and transuranic elements. The process will then be repeated until you hit Ctrl-D (or whatever terminates input on your system). At each stage, only splittings arising at that stage are shown by dots. Atoms that split apart at an earlier stage are separated by an ampersand. When a common or transuranic element first appears, its chemical symbol is shown; it is then removed before computing the next generation. Figure 1 shows the output of

```
 0: 333222112
 1: 33322112
 2: 33222112
 3: 23322112
 4: 1223222112
 5: 112213322112
 6: 21221123222112
 7: 121122211213322112
 8: 111.H.13221121123222112
 9: 31 & 1113222112211213322112
10: 1311 & 311332212221121123222112
11: 111321 & Pm.Ca.32211322112211213322112
12: 31131211 & 13222113222122221121123222112
13: 132113111221 & 11133221133211322112211213322112
14: 1113122113312211 & 3123222.Ca.Li
15: Er.Ca.Sb & 1311121332
16: 11133112112.Zn
17: Zn.321122112
18: 131221222112
19: 1113112211322112
20: 311321222113222112
21: 1321131211322113322112
22: 1113122211311122113222.Na
23: 3113112222133122211332
24: Ho.Pa.H.Ca.Ac.H.Ca.Zn
```

FIGURE 1. The evolution of Thuselum.

the program for the element Thuselum referred to in [L]. The final digit 2 can be replaced by any greater digit.

5. The program `table`

This prints out the periodic table. The information is the same as in [C] except for the abundances (for which see the next section), but arranged slightly differently. Each line shows the atomic number, the chemical symbol, the digit string, and the decomposition of the decay product. Given the option `-p`, all the parents of the element will be printed instead of the daughters, and given the option `-n`, the atomic numbers of the daughters or parents will be given instead of the symbols.

Finally, the option `-m` causes the transition matrix to be printed, while `-M` also suppresses the printing of the table. With either of these, you can also specify a filename, and the matrix will be sent there (while with `-m` the table still goes to standard output). The `-M` option is what I used to produce the data file `mat.txt` for the notebook `mat.nb` to operate on. Note that, as in [C], each row of the matrix gives the decay products of one element (so this matrix operates on the right of row vectors). The ordering of the rows and columns is from 92 (U) down to 1 (H).

The computation of the elements is simple; start with the string 3 and keep decaying and splitting until no new elements are obtained. It is not obvious how to get the right order. Conway arranged things so that each element except the

last has the next amongst its products. However, there are seven such orderings (Bottomley [B]), four starting with the string 3 and three with 3112112. It turns out that using the obvious backtracking algorithm, starting from 3 and testing daughters in left-to-right order, the first ordering found is Conway's. (Perhaps that's how he arrived at it.)

6. The Mathematica notebook `mat.nb`

This rechecks the computations of Conway's constant, its minimal polynomial, and the abundances of the common elements, using the transition matrix computed by `table`. Note that although 50 digits of the constant are given, the computation is only done to 50 digit precision; I don't know enough about the workings of Mathematica to say how many digits of the result can be trusted. It also computes the earliest day by which, starting from any common element, all common elements must be simultaneously present.

References

[B] Henry Bottomley, *Seven complete sequences for the Conway Look and See elements*, `http://www.btinternet.com/~se16/js/lands3.htm`

[C] J.H. Conway, *The weird and wonderful chemistry of audioactive decay*, in: *"Open Problems in Communication and Computation"*, T.M. Cover and B. Gopinath, eds., Springer, 1987, pp. 173–188.

[EZ] Shalosh B. Ekhad and Doron Zeilberger, *Proof of Conway's Lost Cosmological Theorem*, Electron. Res. Announc. Amer. Math. Soc. **3** (1997), 78–82; available at `http://www.math.temple.edu/~zeilberg/mamarim/mamarimhtml/horton.html`.

[L] R.A. Litherland, *Conway's Cosmological Theorem*, `http://www.math.lsu/~lither/jhc/cct.pdf`.

Department of Mathematics, Louisiana State University, Baton Rouge, LA 70803
E-mail address: `lither@math.lsu.edu`