Computer output

DANIEL N. OSHERSON*
MIT Center for Cognitive Science

Hal Hackir was looking at an IBM-BLOKBUSTR, the latest in digital computation. The instruction manual had invited him to get acquainted with his new BLOKBUSTR by loading up and running a program named "K-BAR."

Hal smiled.

"K" is the usual abbreviation for the set of numbers that code Turing machines that converge on their own code numbers. "K-BAR" denotes the complement of K. Years ago Hal had proved as an exercise in an expensive textbook that K-BAR cannot be recursively enumerated by any Turing machine. And since every digital computer is modeled by some Turing machine, Hal knew full well that BLOKBUSTR could not enumerate the members of K-BAR.

Hal turned the page of the BLOKBUSTR instruction manual In the center of an otherwise blank sheet sat the following message:

> No really! Go ahead and try our K-BAR program. There's nothing to lose (is there?).

Hal found himself taken aback by the insistent quality of the manual. But he loaded up the K-BAR program in a few simple steps, and pressed the button marked "run." The BLOKBUSTR teletype began printing:

```
Turing machine code number 0: in K-BAR
Turing machine code number 1: in K-BAR
Turing machine code number 2: in K
Turing machine code number 3: in K-BAR
Turing machine code number 4: in K

              .
              .
              .
```

* Reprint requests should be sent to Dan Osherson, MIT E10-016, 77 Massachusetts Ave., Cambridge, MA 02139, U.S.A.

"Now this is impossible." Hal said to himself. "BLOKBUSTR may be large, lovely, and lightning fast" (as the advertisement put it), "but it *can't* be telling me the members of K-BAR." As he watched the teletype deliver increasing quantities of paper to the office floor, Hal conceived a plan. He would use his old computer (an IBM-POWDRPUF) to simulate Turing machines and produce a recursive enumeration of K (he knew this to be computationally possible). Sooner or later, POWDRPUF might output a member of K that BLOKBUSTR claimed to be in K-BAR. In that case he would write to IBM, documenting their error. Perhaps they would be impressed by his initiative and mathematical acumen ...

Hal's procedure was a complete success: numbers enumerated by the POWDRPUF program for K kept showing up in BLOKBUSTR's list for K-BAR. Moreover, in examining BLOKBUSTR's output, Hal discovered a subtle printing error. In some of the lines, one of the letters was slightly smeared by the printer; there was never more than one smeared letter per line, and many of the lines had no smeared letters at all; there seemed to be no pattern to which letters got smeared.

"Random or systematic effect?" Hal wondered. To find out, he reloaded the K-BAR program and ran it from the beginning. There was no doubt about it. The smearing was systematic, occurring at the same places as before. For later reference, Hal noted the line numbers at which the smearing occurred: 0, 1, 3, 8, 29, 102, ...

But then Hal noticed something surprising. None of the numbers he was writing had appeared as yet in POWDRPUF's enumeration of K. What's more, the gaps between the numbers given by BLOKBUSTR's smear-enumeration kept getting filled up by POWDRPUF's K-enumeration.

Hal began to chuckle, but ceased when he noticed that he was perspiring.

"I'm just going to wait this out," Hal said aloud. "I'll keep both BLOKBUSTR and POWDRPUF cranking away until these two sets of numbers begin to overlap." So he waited. And waited. BLOKBUSTR's smear-enumeration continued to look like an enumeration of K-BAR. To distract himself from the lightheadness that was growing stronger as the evening wore on, Hal turned to page 3 of the BLOKBUSTR instruction manual. There he found the following.

> Surprised, aren't you? Actually, there is nothing supernatural about the K-BAR program. Simply, BLOKBUSTR is both a digital computer, and something else besides. As a computer, BLOKBUSTR can simulate arbitrary Turing machines for arbitrary amounts of time (provided that you give it sufficient blank magnetic tape to use in these computations); and in its computer mode, BLOKBUSTR can do no more than simulate Turing machines, such limitations being the logical fate of any system of digital computation.

"But IBM's patented *smear-a-letter* feature also allows BLOKBUSTR to enjoy a noncomputational mode. IBM engineers have discovered that among the countless functions implemented by lawful processes in the natural world, only very few are Turing computable, even when we digitalize in various ways the range of these real-valued functions. Indeed, the existence of such uncomputable but physically realized functions has been a plausible supposition for many years, since most number-theoretic functions are not computable. Now, noncomputational physical processes involved in the magnetic fields established by BLOKBUSTR's power source implement various uncomputable functions, among them, the characteristic function for K-BAR. IBM engineers have harnessed these functions to smear letters in a potentially informative way—the details of which occupy pages 527–841 of this manual.

"However, before digging into the conventions governing BLOKBUSTR's smear-a-letter capacity, we'd like to take this opportunity to answer charges leveled at BLOKBUSTR by certain small, rival companies operating in outlying, coastal regions of the United States. (They have even admitted to launching a 'smear campaign' against us.) The charge is that BLOKBUSTR is not a genuine digital computer because it exploits physical processes that help determine its output. To this we make two replies.

"First, *every computer* exploits physical processes for the purpose of doing even the most ordinary calculations; how else could a computer—which is, after all, just a physical object—work? Second, every digital computer makes responses beyond the established user-recognized repertoire: every computer undergoes detectable temperature changes, field shifts, static discharge, and so forth. Hence, *all computers* potentially code uncomputable functions in their behavior. BLOKBUSTR simply harnesses these behaviors in a way that makes them informative to the user. Part of that process involves digitalizing the output of several physical variables (see Appendix 37, Part 16 for details).

"Is BLOKBUSTR a digital computer? It all depends on how you construe its outputs. If you construe its output one way (as printed numerals), then BLOKBUSTR is a *bona fide* digital computer. If you construe its output another way (as smear-tokens), then BLOKBUSTR is definitely not a digital computer.

"Now, dear reader, if you find yourself upset by the ambiguity inherent in BLOKBUSTR's dual mode of operation, ask yourself whether or not *you* are a digital computer. When you do proofs in formal arithmetic, you can remember only a finite number of axioms. Hence, the theorems you can prove constitute an effectively enumerable set. But what if you're willing to construe your output not as the mere sequence of words you utter (ignoring here complexities about the nature of words), but as words along with slight hesitations, tremulations in pitch, nervous ticks, and what not—in short, along with the multitude of physically based effects on your behavior that arise from the electrical and chemical activity of your body? Would you not expect certain uncomputable functions to be thereby implemented—even in a stimulus-free, sealed room, so that all your behavior is endogenously generated? Well then. Are you or are you not a digital computer? It seems to depend entirely on how you construe your output!

"Furthermore, ...

But Hal closed the manual at this point. He was feeling depressed. BLOKBUSTR and POWDRPUF continued to pile output onto the office floor, apparently providing the characteristic function for K-BAR (depending on how you look at it).

"Am I, or am I not a digital computer?" Hal mused. He absently perused the books, manuals, lab reports, and journals that littered his office. Eventually, he came upon this very story.