COMPUTABILITY BY NORMAL ALGORITHMS

R. M. BAER

1. **Introduction.** In this note we give a short proof of the normal Markov algorithmic computability of partial recursive functions. Moreover, the target strings occurring during computations are of a very simple kind. Detlovs [1] proved the equivalence of the class of partial recursive functions and the class of normal algorithmically computable functions. The proof, below, that the partial recursive functions are normal algorithmically computable depends upon the well-known result due to Minsky (see [2], [3]) that if f is a partial recursive function (of k variables) then there is a program P which, starting with an initial target number $u = p_1^{x_1} \cdot \cdot \cdot p_k^{x_k}$ (where the p_i are distinct primes and, below, p may range over a finite set of primes), transforms u into $z = p_1^{f(x_1, \cdots, x_k)}$ if $f(x_1, \cdot \cdot \cdot, x_k)$ is defined; P never terminates if $f(x_1, \cdot \cdot \cdot, x_k)$ is undefined. Moreover, P consists of a (finite) list of instructions I_α which have the form I_α: $M(p)$, β or I_α: $D(p)$, β, γ where $M(p)$, β means multiply the current target number by p and continue with instruction I_β; $D(p)$, β, γ means divide the current target number by p (if the number is indeed divisible by p) and continue with instruction I_β, or, in the event that the current target number is not divisible by p, leave the number intact and continue with instruction I_γ.

2. **Proof of normal computability.** Let f be a partial recursive function and let $P = (I_1, \cdot \cdot \cdot, I_{n_f})$ be a program which computes f. We take $A = \{0, 1\}$ as a two-element alphabet and construct, over A, a normal algorithm $\mathfrak{N}_P$ which (in an obvious way) is equivalent to P. We say that a string over A of the form $0^h1^j0^k$ $(h, j, k \geqq 1)$ is *dihedral.* It will be seen that the algorithm $\mathfrak{N}_P$ is applied to, and produces only, dihedral words. We represent ordered pairs $\langle m, n \rangle$ of natural numbers by dihedral words of the form $0^{m+1}1^\alpha 0^{n+1}$ $(\alpha \geqq 1)$ and we use the substring $01^\alpha 0$ as an instruction marker. Then to construct $\mathfrak{N}_P$ we provide, for each instruction I_α in P, a corresponding normal subalgorithm $\bar{I}_\alpha$ which duplicates the action of I_α. Thus, corresponding to

$$I_\mu: M(p), \beta \quad \text{and} \quad I_\delta: D(p), \beta, \gamma,$$

Received by the editors October 24, 1967.

we define:

$$\bar{I}_\mu \begin{cases} 001^\mu 0 \to 01^\mu 0^{p+1} \\ 01^\mu 00 \to 001^\sigma 0 \\ 01^\sigma 00 \to 001^\sigma 0 \\ 01^\sigma 0 \to 01^\beta 0 \end{cases} \qquad \bar{I}_\delta \begin{cases} 0^{p+1}1^\delta 0 \to 01^\nu 00 \\ 0^2 1^\delta 0 \to 0^2 1^\gamma 0 \\ 0^{p+1}1^\nu 0 \to 01^\nu 00 \\ 0^p 1^\nu 00 \to 0^{2p}1^\sigma 0 \\ 0^{p-1}1^\nu 00 \to 0^{2p-1}1^\sigma 0 \\ \vdots \\ 0^2 1^\nu 00 \to 0^{p+2}1^\sigma 0 \\ 01^\sigma 00 \to 0^{p+1}1^\sigma 0 \\ 01^\sigma 0 \to 01^\gamma 0 \\ 01^\nu 00 \to 001^\kappa 0 \\ 01^\kappa 00 \to 001^\kappa 0 \\ 01^\kappa 0 \to 01^\beta 0 \end{cases}$$

If the superscripts are chosen suitably distinct, it is clear that each such $\bar{I}_\mu$ and $\bar{I}_\delta$ operates as required and that the sequence of instructions $\bar{I}_\alpha$ followed in $\mathfrak{N}_P$ corresponds exactly to the sequence of instructions followed in P when $\mathfrak{N}_P$ and P start with corresponding initial data. It then follows that $\mathfrak{N}_P$, when applied to the initial target word $0^{u+1}10$ ($u = p_1^{x_1} \cdots p_k^{x_k}$), fails to terminate precisely when $f(x_1, \cdots, x_k)$ is not defined; if $z = f(x_1, \cdots, x_k)$ is defined, then for some τ, the computation ends with the word $0^{v+1}1^\tau 0$ where $v = p_1^z$.

References

1. V. K. Detlovs, *The equivalence of normal algorithms and recursive functions*, Trudy Mat. Inst. Steklov **52** (1958), 75–139.

2. M. L. Minsky, *Recursive unsolvability of Post's problem of "tag" and other topics in the theory of Turing machines*, Ann. of Math. **74** (1961), 437–455.

3. J. C. Shepherdson and H. E. Sturgis, *Computability of recursive functions*, J. Assoc. Comput. Mach. **10** (1963), 217–255.

University of California, Berkeley