Ilie Petre-Cristian

Grupa 333

Tema CC - Seminar 7

Varianta 1:

1) Descrieti o masina Turing cu mai multe benzi care primește ca input un numar natural nenul n si accepta inputul daca n este compus.

   Initial Banda 1 contine numarul n reprezentat in baza 1 (fara 1-ul in plus)

Pasi:

1. Pe Banda 2 scriem numarul 2 => O(1)
2. Incepem sa ne miscam pe Banda 1 si Banda 2 simultan la dreapta.
3. Daca se ajunge la Blank doar pe a doua banda, se reia banda 2 de la capat si se continua pasul 2
4. Daca se ajunge la Blank doar pe prima banda, se mai adauga un 1 pe banda 2 (creste numarul cu o unitate), se verifica daca numarul de pe banda 2 este inca mai mic decat radical din cel de pe banda 1 (se deplaseaza de k ori k spatii pe prima banda unde k este numarul de pe banda 2, daca se ajunge pe un blank, am depasit sqrt(n)) , se reia banda 1 de la inceput si apoi se reia pasul 2 de la capat.
5. Daca se ajunge la Blank simultan, se accepta cuvantul. (am gasit un numar care, inmultit cu cel de pe banda 2 va da n) -> Pentru fiecare plimbare pe banda avem O(n), facem maxim sqrt(n)

verificari => $O(n * \sqrt{n})$. Pentru verificarea faptului ca numarul de pe a doua banda depaseste sqrt(n) se mai utilizeaza inca un O(n)

Complexitate spatiu: Prima banda O(n), a doua banda are maxim sqrt(n) => O(n + sqrt(n))

Complexitate timp: O(1 + n*sqrt(n) + n*sqrt(n)) ~= O(n * sqrt(n))

2) Descrieti o masina Turing cu o singura banda care primeste ca input numerele naturale nenule n, x si p scrise in baza 1 (fara 1-ul in plus), unde x este putere a lui n, numerele sunt delimitate pe banda prin cate un simbol 0 si care calculeaza functia log_n (x^p).

Log_n(x^p) = p * log_n(x) unde x poate fi scris ca n^k => Problema poate fi redusa la aflarea lui p * log_n (n^k) = p * k

Initial banda arata: Bn0x0pB = Bn0n^k0pB

Pasi:

1. Scriem 0 in stanga lui n si in dreapta lui p. Scriem p in dreapta lui 0. => O(n + x + p)
2. Impartim al doilea numar la primul astfel: marcam pe rand 1 din primul si al doilea numar, cand primul numar este marcat in intregime, il demarcam si scriem in stanga, dupa 0, un 1. Repetam procesul pana ce al doilea numar este marcat in intregime Bn^(k-1)0n0n^k0pB => O(n*(n^k)) ~= O(n^k+1)
3. Verificam daca numarul din stanga (noul numar 1) este egal cu n, daca nu, adunam p la ultimul numar (marcam de la dreapta cate un 1, scriem 1 in dreapta celor marcati, la final demarcam tot) si

incepem sa calculam primul numar impartit la n, rezultatul stocandu-l peste vechiul n^k (dupa ce golim acel spatiu) $Bn^{(k-1)}0n0n^{(k-2)}0\ 2*p\ B \Rightarrow O(2*n + x + (p*(p+1))/2 + n^k) \sim=$
$= O(p^2 + n^k)$

4. Continuam etapele 2-3 pana ce primul sau al treilea numar este egal cu n. Cand unul din ele este egal cu n => ultimul numar va fi egal cu $p*k$

Complexitate spatiu: $O(n + x + n^{(k-1)} + (k+1)*p)$

Complexitate timp: $O(n + x + p + k*(n^k + p^2))$