Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 23, n° 1, 1989, p. 101 à 111)

ON COMPUTATIONS WITH INTEGER DIVISION

by Bettina JUST (1), Friedhelm MEYER AUF DER HEIDE (2) and Avi WIGDERSON (3)

Abstract. – *We consider computation trees (CT's) with operations $S \subset \{+, -, *, \mathrm{DIV}, \mathrm{DIV}_c\}$, where DIV denotes integer division and DIV_c integer division by constants. We characterize the families of languages $L \subset \mathbb{N}$ that can be recognized over $\{+, -, \mathrm{DIV}_c\}$ and $\{+, -, *, \mathrm{DIV}\}$, resp. and show that they are identical. Furthermore we prove lower bounds for CT's with operations $\{+, -, \mathrm{DIV}_c\}$ for languages $L \subset \mathbb{N}$ which only contain short arithmetic progressions. We cannot apply the classical component counting arguments as for operation $S \subset \{+, -, *, /.\}$ because of the DIV_c-operation. Such bounds are even no longer true. Instead we apply results from the Geometry of Numbers about arithmetic progressions on integer points in high-dimensional convex sets for our lower bounds.*

Résumé. – *On considère des arbres étiquetés par un ensemble S d'opérations S contenu dans $\{+, -, *, \mathrm{DIV}, \mathrm{DIV}_c\}$ où DIV représente la division des entiers et DIV_c la division des entiers par une constante. On caractérise les familles de langages L de $\mathbb{N}$ qui peuvent être reconnues à l'aide de $\{+, -, \mathrm{DIV}_c\}$ et $\{+, -, *, \mathrm{DIV}\}$ respectivement et on montre que ces deux classes sont identiques. De plus on donne des bornes inférieures pour les arbres étiquetés par $\{+, -, \mathrm{DIV}_c\}$ pour les langages L de $\mathbb{N}$ qui ne contiennent que des progressions arithmétiques courtes. On ne peut plus appliquer les arguments classiques de comptage comme pour les ensembles d'opérations contenus dans $\{+, -, *, /.\}$ à cause de l'opération DIV_c. A la place, on utilise pour établir les bornes inférieures des arguments de géométrie des nombres sur les progressions arithmétiques de points à coordonnées entières dans des espaces convexes de grande dimension.*

1. INTRODUCTION

The most common operations on integers supported by classical programming languages are $+, -, *, \mathrm{DIV}$, where DIV denotes integer division. In this paper we examine what can be computed with these operations and how efficiently it can be done.

(1) Fb Mathematik, Johann Wolfgang Goethe Universität, 6000 Frankfurt a.M., F.R.G.
(2) Fb Informatik, Universität Dortmund, 4600 Dortmund, F.R.G., supported in part by the Deutsche Forschungsgemeinschaft, ME 872/1-1 and WE 1066/1-2, and the Leibniz Center for Research in Computer Science.
(3) Computer Science Department, Hebrew University, Jerusalem, Israel.

Informatique théorique et Applications/Theoretical Informatics and Applications
0296-1598 89/01 101 11/$3.10/

For this purpose we consider computation trees (CT's) in which operations from some set $S \subset \{+, -, *, \text{DIV}, \text{DIV}_c\}$ can be applied to inputs, arbitrary rational constants, and previously computed values to compute functions $f(x)$ of the input $x \in \mathbb{N}$. DIV_c denotes inhegt division by constants. Furthermore one can execute branchings according to "$f(x) > 0$". There is a variety of papers dealing with operation sets $S \subset \{+, -, *, ./.\}$, [1, 4, 6, 9]. The lower bounds achieved there apply methods from Algebraic Geometry to bound the number of connected components a language $L \subset \mathbb{R}^n$ recognized by such a CT of depth T can have. This is known as the component counting lower bound. In [2] and [6] results are also carried over to languages $L \subset \mathbb{N}^n$. The arguments from Algebraic Geometry work there because only "nice" functions, namely rational functions, can be computed over $\{+, -, *, ./.\}$.

One can easily apply the above results to get a characterization of the families of languages $L \subset \mathbb{N}$ that can be recognized by CT's over $\{+, -\}$ and $\{+, -, *\}$, resp.:

Both families are the same, namely all $L \subset \mathbb{N}$ where L or $\mathbb{N}$-L is finite.

If we now add the DIV-operation, things become much more difficult. For example, we now can recognize languages $L \subset \mathbb{N}$ where L and $\mathbb{N} \setminus L$ are infinite, e. g. arithmetic progressions $\{a + \lambda d, \lambda \in \mathbb{N}\}$ for some d, $a \in N$. This can be done by the test

$$d \cdot ((x-a) \text{DIV}_c d) \doteq x - a.$$

Note that we can express the above even without multiplication because d is a constant.

In section III, after having introduced the computation models in section II, we present a characterization of those languages $L \subset \mathbb{N}$ that can be recognized over $\{+, -, \text{DIV}_c\}$ and $\{+, -, *, \text{DIV}\}$, resp..

It turns out that the two families of languages are identical. They consist of all languages of the type $B \cup \{a + \lambda d, d \in A, \lambda \in \mathbb{N}\}$ with $a \in \mathbb{N}$, A, $B \subset \mathbb{N}$, A, B finite.

We call them AP-languages, AP stands for arithmetic progression.

A much more general model of CT's is considered in [2]. Here the application of arbitrary analytic functions and DIV are allowed operations. One still gets results on what cannot be computed. One example is the solvability of linear diophantine equations. Upper bounds for approximate solutions for this problem can be found in [5], lower bounds dependent on the binary input size in [9].

In section IV we prove lower bounds for CT's over $\{+, -, \mathrm{DIV}_c\}$, the weakest model that can recognize AP-languages. For operations $\{+, -, *, ./.\}$, it is shown in [1, 9] that a lower bound of $\Omega(\log(q))$ holds for recognizing $L \subset \mathbb{N}$, if L consists of q intervals $\{i, i+1, \ldots, j\}$. Such easy characterizations of "hard" languages are no longer true if also DIV_c or DIV is allowed.

We show the following lower bound:

If $L \subset \mathbb{N}$ has n elements and does not contain any arithmetic progression of length $k+1$, then each CT over $\{+, -, \mathrm{DIV}_c\}$ for L needs $\Omega(\log(n)/\log\log(n))$ steps, if $k \leqq \log(n)$, and $\Omega(\log(n)/\log(k))$ steps else.

We give several examples. The lower bound is true e. g. for $\{2^i, i=1, \ldots, n\}$, because this language does not contain any arithmetic progression of length 3.

To prove the lower bound we apply methods from the Geometry of Numbers, based on results from [7, 8].

We show (lemma 3):

If a convex set in $\mathbb{R}^n$ contains $\Omega(n^{\Omega(n)} \cdot k^n)$ integer points, then at least $k+1$ of them are on one straight line.

We conjecture that the above result is even true if the number of integer points is $\Omega(k)^{\Omega(n)}$. This would yield asymptotically optimal lower bounds, e. g. $\Omega(\log(n))$ for recognizing $\{2^i, i=1, \ldots, n\}$ over $\{+, -, \mathrm{DIV}_c\}$.

It looks like a very complicated task to prove any lower bound for an AP-language on CT's over $\{+, -, *, \mathrm{DIV}\}$, because the structure of functions that can be computed with these operations seems to be very difficult.

II. THE COMPUTATION MODELS

A Computation Tree with operation set $S \subset \{+, -, *, \mathrm{DIV}, \mathrm{DIV}_c\}$ ($S-$CT) for n inputs $x_1, \ldots, x_n \in \mathbb{N}$ is a rooted tree with degrees from $\{0, 1, 2\}$. Nodes with degree 0, the leaves, are either accepting or rejecting. Nodes v with degree 1 are labelled with a function $g_v: \mathbb{N}^n \to \mathbb{Q}$, $g_v = f_1 \circ f_2$ with $\circ \in S$, f_1, f_2 either rational constants, or input variables or functions previously computed on the path to v. Nodes v with degree 2 are labelled with predicates "$p(x_1, \ldots, x_n) > 0$" for some function p previously computed on the path to v. An input $(x_1, \ldots, x_n) \in \mathbb{N}^n$ follows a path in the tree defined by the outcomes of the predicates (True $\hat{=}$ "go left", False $\hat{=}$ "go right") and is accepted, if its path arrives at an accepting

leaf. Thus an S-CT recognizes a language $L \subset \mathbb{N}^n$. The complexity of an S-CT is its depth.

For technical reasons, in order to prove theorem 1, we need a somewhat artificial type of computation trees.

A modulo-Branching Tree (MBT) is a $\{+, -, *\}$-CT for one input $x \in \mathbb{N}$ which contains additional branchings of arbitrary, finite degree. If the input x arrives at a node with degree β, then x follows the i'th branch, $i \in \{0, \ldots, \beta-1\}$, iff $x \bmod \beta = i$.

Again an input x follows one path in the tree and is accepted if it arrives at an accepting leaf. The complexity of the tree is its depth.

III. COMPUTABILITY WITH $\{+, -, \mathrm{DIV}_c\}$-CT's AND $\{+, -, *, \mathrm{DIV}\}$-CT's

In this section we show that the families of languages $L \subset \mathbb{N}$ that can be recognized by $\{+, -, \mathrm{DIV}_c\}$-CT's and $\{+, -, *, \mathrm{DIV}\}$-CT's, resp., are identical, namely the AP-languages. (AP stands for arithmetic progression.)

DEFINITION: Let $a \in \mathbb{N}$, $A, B \subset \mathbb{N}$, A, B finite.

$$L(a, A, B) := B \cup \{d + \lambda a, \lambda \in \mathbb{N}, d \in A\}.$$

Such languages $L(a, A, B)$ are called AP-languages.

Now we are ready to state the main result of this section.

THEOREM 1: *Let $L \subset \mathbb{N}$. Then the following four statements are equivalent.*

(i) *L is an* AP*-language.*

(ii) *L can be recognized by an* MBT.

(iii) *L can be recognized by a* $\{+, -, \mathrm{DIV}_c\}$-CT.

(iv) *L can be recognized by a* $\{+, -, *, \mathrm{DIV}\}$-CT.

In particular, the families of languages $L \subset \mathbb{N}$, that can be recognized by $\{+, -, \mathrm{DIV}_c\}$-CT's and $\{+, -, *, \mathrm{DIV}\}$-CT's *are identical, namely the* AP*-languages.*

Proof: We show (i)⇒(iii)⇒(iv)⇒(ii)⇒(i).

The main part is the proof of (iv)⇒(ii).

(i)⇒(iii):

Let $L=L(a, A, B)$ be an AP-language. $A\{+, -, \mathrm{DIV}_c\}$-CT T for L works as follows:

– T first checks whether $x\in B$ by binary search. If yes, we are ready.

– Otherwise:

T checks for each $d\in A$, whether $x\in\{d+\lambda a, \lambda\in\mathbb{N}\}$. This can be done by the test "$a\cdot((x-d)\,\mathrm{DIV}_c\, a)=x-d$". Note that a is a constant, therefore $a\cdot((x-d)\,\mathrm{DIV}_c\, a)$ can be computed without multiplication. □

(ii)⇒(iv) is trivial.

(ii)⇒(i):

Consider an MBT T recognizing some language $L\subset\mathbb{N}$. Let v be an accepting leaf v of T, where an input set $c(v)$ arrives. The binary branchings on the path to v add restrictions of the form "$p(x)>0$" or "$p(x)\leqq 0$", where p is a polynomial in x. (Other functions cannot be computed with the operations $+, -, *$.) Thus $c(v)$ is either finite or $c(v)$ can be represented as $B_v\cup I_v$. Here B_v is a finite set containing all elements of $c(v)$ belonging to bounded connected components of sets $\{x, p(x)>(\leqq)0\}$, coming from binary branchings. I_v is of the form $I_v=\{x \mid x>\beta_v,\ x \bmod \delta_j=i_j \text{ for } j=1,\ldots,r\}$, where the r high degree branchings on the path to v have degrees $\delta_1,\ldots,\delta_r$, at the jth such branching, the i_j's branch is chosen by the path, and $\beta_v=\max B_v$.

It is well known that I_v can be expressed as one arithmetic progression,

$I_v=\{d_v+\lambda a_v, \lambda\in\mathbb{N}\}$ for suitable d_v, a_v. Let V be the set of those accepting leaves where infinitely many inputs arrive.

Then $I=\bigcup_{v\in V} I_v$ is the union of finitely many arithmetic progressions.

Again it is a well known fact from combinatorics that I can be represented as $B'\cup\{d+\lambda a, d\in A\}$ for some $a\in\mathbb{N}$ and finite sets B' and A.

Let now B'' be the finite set of inputs arriving at those accepting leaves where only finitely many inputs arrive, $B''':=\bigcup_{v\in V} B_v$, $B:=B'\cup B''\cup B'''$. Then we have that $L=L(a, A, B)$. Thus L is an AP-language. □

(iv)⇒(ii):

This is the main part of the proof.

Let T be $a\{+, -, *, \mathrm{DIV}\}$-CT recognizing $L\subset\mathbb{N}$. We shall show how to replace DIV-operations by high degree branchings such that the resulting MBT recognizes a languages L' which is identical to L for sufficiently large

x. For small x we again can simply add a binary search procedure in order to get an MBT for L.

The key observation which yields the desired result is formulated in the next lemma.

LEMMA 1: *Let p, $q: \mathbb{N} \to \mathbb{Q}$ be polynomials with rational coefficients, degree $(p) \geqq$ degree (q).*

Then there are β, $z \in \mathbb{N}$, such that for each $i \in \{0, \ldots, \beta-1\}$ there is a polynomial $r_i: \mathbb{N} \to \mathbb{Q}$ with rational coefficients such that $p(x)$ DIV $q(x) = r_i(x)$ for all $x \geqq z$ with $x \bmod \beta = i$.

Before we prove this lemma, we finish the proof of "(iv)⇒(ii)".

Let v be a first node on a path in T where a DIV-operation is executed. Then a function $p(x)\,\mathrm{DIV}\,q(x)$ for two previously computed polynomials p, q with rational coefficients is computed. If degree $(p) <$ degree (q) then we replace $p(x)\,\mathrm{DIV}\,q(x)$ by the constant 0. This is correct for sufficiently large x. Assume now that degree $(p) \geqq$ degree (q).

Let β, z be chosen as in the lemma. Then we replace v by a degree-β-branching. At the i-th branch, $i \in \{0, \ldots, \beta-1\}$, we attach a computation over $\{+, -, *\}$ for the polynomial r_i from the lemma. Below we place of copy of the subtree below v from T and replace each use of $p(x)\,\mathrm{DIV}\,q(x)$ as an operand in this subtree by $r_i(x)$. By the lemma the resulting computation tree recognizes a language $L' \subseteq \mathbb{N}$ with $L' \cap \{x \in \mathbb{N},\ x \geqq z\} = L \cap \{x \in \mathbb{N},\ x \geqq z\}$.

By the same procedure we replace step by step all DIV-instructions, and finally we come up with an MBT recognizing a language L'' with $L'' \cap \{x \in \mathbb{N},\ x \geqq z'\} = L \cap \{x \in \mathbb{N},\ x \geqq z'\}$ for some sufficiently large z'. Finally, in order to recognize L, we first test whether $x \geqq z'$. If yes we use the above MBT, otherwise we apply a binary search procedure in order to recognize the finite language $L \cap \{x \in \mathbb{N},\ x \leqq z'\}$. □

In order to finish the proof we have to present a:

Proof of lemma 1: By elementary algebra we know that there are polynomials r, $s: \mathbb{N} \to \mathbb{Q}$ with rational coefficients, degree$(s) <$ degree (q), such that $p = r \cdot q + s$. Choose $\beta \in \mathbb{N}$ such that there is a polynomial $\tilde{r}: \mathbb{N} \to \mathbb{Q}$ with integer coefficients and $r = 1/\beta\,\tilde{r}$. Then we have:

($\lfloor a \rfloor$ denotes the largest integer smaller or equal to a.)

$$p(x)\,\mathrm{DIV}\,q(x) = \left\lfloor r(x) + \frac{s(x)}{q(x)} \right\rfloor = \left\lfloor \frac{1}{\beta}\tilde{r}(x) + \frac{s(x)}{q(x)} \right\rfloor$$

Because degree $(s)<$ degree (q), $\lim_{x\to\infty} s(x)/q(x)=0$.

Therefore, for sufficiently large x, $p(x)\,\mathrm{DIV}\,q(x)=r(x)\,\mathrm{DIV}\,\beta$. Now let $i\in\{0,\ldots,\beta-1\}$ be fixed, and x mod $\beta=i$, i.e. $x=\lambda\beta+i$ for some $\lambda\in\mathbb{N}$.

Let $\tilde{r}(x)=\sum_{j=0}^{n} a_j x^j$, $a_j\in\mathbb{Z}$.

If we consider i as a constant, we can write $\tilde{r}(x)=\tilde{r}(\beta\lambda+i)$ as a polynomial in λ which has the form

$$\tilde{r}(x)=\sum_{j=0}^{n} b_j(\lambda\beta)^j \qquad \text{with} \quad b_j\in\mathbb{Z} \quad (b_j \text{ may depend on } i).$$

This yields:

$$\tilde{r}(x)=\sum_{j=0}^{n} b_j(\lambda\beta)^j=b_0+\beta\cdot g(\lambda) \text{ with } g(\lambda)=\sum_{j=1}^{n} b_j\beta^{j-1}\lambda^j\in\mathbb{Z}$$

Thus we obtain, for sufficiently large x with x mod $\beta=i$:

$p(x)\,\mathrm{DIV}\,q(x)=\tilde{r}(x)\,\mathrm{DIV}\,\beta=b_0\,\mathrm{DIV}\,\beta+g(\lambda)$

$=b_0\,\mathrm{DIV}\,\beta+g((x-i)/\beta)=$ polynomial in x, which proves lemma 1. □

III. LOWER BOUNDS FOR $\{+,-,\mathrm{DIV}_c\}$-CT's

THEOREM 2: *Let $L\subset\mathbb{N}$, $\#L=n$. If L contains no arithmetic progression of length $k+1$, then each $\{+,-,\mathrm{DIV}_c\}$-CT for L has complexity $\Omega(\log(n)/\log\log(n))$, if $k\leq\log(n)$, and $\Omega(\log(n)/\log(k))$ else.*

Examples: – $L_n:=\{2^i,\ i=1,\ldots,n\}$ has n elements and no arithmetic progression of length 3.

Therefore a lower bound $\Omega(\log(n)/\log\log(n))$ holds.

A more general example covers all relations between k and n.

– $L_{l,k}:=\{j\cdot(k+1)^i,\ i=0,\ldots,l-1,\ j=1,\ldots,k\}$ has $n=l\cdot k$ elements and no arithmetic progression of length $k+1$. Therefore the lower bound $\Omega(\log(n)/\log(k))$ holds.

Proof of theorem 2: Consider a $\{+,-,\mathrm{DIV}_c\}$-CT T with depth D recognizing a finite language $L\subset\mathbb{N}$, which contains no arithmetic progression of length $k+1$, $\#L=n$. Let v be a leaf of T, $v_1,\ldots,v_d=v$ be the path to v,

$d \leqq D$. Let $f_i: \mathbb{N} \to \mathbb{Q}$ denote the function computed at v_i, and $c(v)$ the set of inputs arriving at v.

The following lemma presents a characterization of $c(v)$.

LEMMA 2: *There is a convex polytope P in $\mathbb{R}^{d+1}$ such that the following holds:*

(i) $x \in c(v) \Leftrightarrow \exists (c_1, \ldots, c_d) \in \mathbb{Z}^d : (x, c_1, \ldots, c_d) \in P$.

(ii) *For each $x \in c(v)$ there is exactly one $(c_1, \ldots, c_d) \in \mathbb{Z}^d$ with $(x, c_1, \ldots, c_d) \in P$.*

Proof: We replace, from top to bottom, each DIV_c-operation by a new variable c_j. We need at most d new variables. Whenever the result of a DIV_c-operation is used as an operand, we use the associated new variable instead. Thereby, at each node v_i a function $g_i(x, c_1, \ldots, c_d)$ is computed. The g_i's are linear, because only operations $+$, $-$ are used for their computations. Let $I \subset \{1, \ldots, d\}$ be the set of indices i such that, at v_i, DIV_c is applied. Then we define the restrictions:

$$(*) \qquad b_i \cdot c_i \leqq (\geqq)\, a_i(x, c_1, \ldots, c_d) < (>)\, b_i(c_i + 1)$$

where $a_i(x, c_1, \ldots, c_d)\, \mathrm{DIV}_c\, b_i$ is computed at v_i, $i \in I$. The choice of $(\leqq, <)$ or $(\geqq, >)$ depends on whether b_i is positive or negative.

We now add the appropriate restrictions

$$(**) \qquad g_i(x, c_1, \ldots, c_d) > (\leqq) 0 \text{ for all branching nodes } v_i$$

to the system of inequalities from $(*)$, where $>$, $\leqq$ is defined according to the branch chosen.

One easily verifies that the solution set P of the system of linear inequalities from $(*)$ and $(**)$ fulfills (i) and (ii). P is a (convex) polytope. □

Let P_v be the polytope associated with the path to v. Note that lemma 2 implies that $\#P_v \cap \mathbb{Z}^{d+1} = \#c(v)$. We now shall see that $\#P_v \cap \mathbb{Z}^{d+1}$ is small, if $c(v)$ does not contain a long arithmetic progression. For this purpose let us call the integer points on a line segment in a convex set P a progression in P.

LEMMA 3: *Let B denote the set of integer points in a convex subset of $\mathbb{R}^n$. If B does not contain a progression of length $(k+1)$, then $\#B \leqq k^n \cdot n^{O(n)}$.*

Before we prove lemma 3 we finish the proof of the theorem. Let v be an accepting leaf. As the recognized language does not contain an arithmetic progression of length $k+1$, $c(v)$ does not, too. Therefore, $B_v := P_v \cap \mathbb{Z}^{d+1}$

contains no progression of length $k+1$. By lemma 3 and the above, we get $\#c(v)=\#B_v \leqq k^{d+1} \cdot d^{O(n)}$. As T has at most 2^D accepting leafs, L has at most $2^D \cdot D^{O(D)} \cdot k^{D+1}$ elements, which yields $n=\#L \leqq 2^D \cdot D^{O(D)} \cdot k^{D+1}$. Solving this inequality proves theorem 2. □

For $p \in \mathbb{R}^n$, $r>0$ let $B(p, r)$ denote the n-dimensional ball with center p and radius r, i.e. $B(p, r) := \{x \in \mathbb{R}^n, \|x-p\| \leqq r\}$, where $\|..\|$ denotes the Euclidean metric.

Furthermore, for a basis $b_1, \ldots, b_n$ of $\mathbb{R}^n$, $S(b_1, \ldots, b_n) := \sum_{i=1}^{n} b_i \mathbb{Z}$ is the lattice with basis $b_1, \ldots, b_n$.

We need the following two lemmas. The first is shown in [8], the second in [7].

LEMMA 4 [8]: *Let S be a lattice in $\mathbb{R}^n$, $p \in \mathbb{R}^n$, $r>0$. If $B(p, r) \cap L = \emptyset$, then there is a hyperplane H in $\mathbb{R}^n$ and a vector $d \in \mathbb{R}^n$ such that $S \subset \bigcup_{\lambda \in \mathbb{Z}} (H + d \cdot \lambda)$ and $B(p, r)$ is contained in the convex hull of $(H + d \cdot \lambda)$ and $(H + d(\lambda + n^{3/2} - 1))$ for some $\lambda \in \mathbb{Z}$. We say for short: $B(p, r)$ is covered by $n^{3/2}$ hyperplanes relative to S.*

LEMMA 5 [7]: *Let P be a convex polytope in $\mathbb{R}^n$. Then there is a nonsingular, linear mapping $\tau: \mathbb{R}^n \to \mathbb{R}^n$, a $p \in \tau(P)$, and radii r, $R>0$, such that:*

(i) $R/r \leqq 2\, n^{3/2}$;

(ii) $B(p, r) \subset \tau(P) \subset B(p, R)$.

Proof of Lemma 3: Let $f(n, k)$ denote the maximum number of integer points a convex set in $\mathbb{R}^n$ without progression of length $k+1$ can contain.

$$\text{Clearly, } f(1, k) = k.$$

Let $n>1$. Let P be the convex hull of B, then P is a convex polytope. Apply lemma 5 to find τ, p, r, R such that (i) and (ii) hold. Let S denote the lattice $S(\tau(e_1), \ldots, \tau(e_n))$ where the e_i's are the unit vectors in $\mathbb{R}^n$. We distinguish between two cases.

Case 1: $B(p, r) \cap S = \emptyset$.

Then, by lemma 4, $B(p, r)$ can be covered by $n^{2/3}$ hyperplanes. Thus, by lemma 5 (i), $B(p, R)$ can be covered by $R/r \cdot n^{3/2} \leqq 2n^3 =: d$ hyperplanes relative to S. As, by lemma 5 (ii), $\tau(P) \subset B(p, R)$ holds, $\tau(P)$ is also covered by them. As τ is nonsingular and linear, the backimages $H_1, \ldots, H_d$ of these hyperplanes contain $P \cap \mathbb{Z}^n = B$.

As B does not contain any progression of length $k+1$, $B \cap H_i$ does not, too. Therefore we obtain: $\#B \leqq 2n^3 \cdot f(n-1, k)$.

(Note the $H_i \cap \mathbb{Z}^n$ can be locked upon as an $(n-1)$-dimensional lattice.)

Case 2: $B(p, r) \cap S \neq \emptyset$

Now consider the lattice $\bar{S} := k \cdot S$.

Then $\#(\bar{S} \cap B(p, r)) \leqq 1$.

To see this assume that x an y, $x \neq y$, are contained in $\bar{S} \cap B(p, r)$.

Then the line segment $[x, y]$ in $\mathbb{R}^n$ would contain $k+1$ points $a_0, \ldots, a_k \in S$. Thus $\tau^{-1}(B(p, r))$, and therefore P, would contain the progression $\tau^{-1}(a_0), \ldots, \tau^{-1}(a_k)$ of length $k+1$, a contradiction.

For simplicity, we assume w.l.o.g. that $\bar{S} \cap B(p, r) = \{k \cdot \tau(e_1)\}$. This can be achieved by a linear translation of P that does not change the structure of its integer points. Now consider the lattice $\tilde{S} := 2k \cdot S = 2 \cdot \bar{S}$. Now, $B(p, r) \cap \tilde{S} = \emptyset$, because $k \cdot \tau(e_1) \notin \tilde{S}$. The argumentation from case 1 now guarantees that $\tau(P)$ can be covered by $2n^3$ hyperplanes relative to $\tilde{S}$. As $\tau^{-1}(\tilde{S}) = (2k\mathbb{Z})^n$, we can conclude that P can be covered by $2n^3$ hyperplanes relative to the lattice $(2k\mathbb{Z})^n$. Thus P can be covered by $c = 2k \cdot 2n^3$ hyperplanes $H_1, \ldots, H_c$ relative to $\mathbb{Z}^n$, i.e., $B \subset \bigcup_{i=1}^{c} H_i$. The argumentation from case 1 now shows that $\#B \leqq c \cdot f(n-1, k) = 4kn^3 \cdot f(n-1, k)$.

The two cases above now imply that $f(n, k) \leqq 4kn^3 \cdot f(n-1, k)$.

Thus $f(n, k) \leqq k^n \cdot n^{O(n)}$ which proves lemma 3.

ACKNOWLEDGMENTS

We would like to thank Martin Dietzfelbinger for valuable discussions about lemma 3.

REFERENCES

1. M. Ben Or, *Lower bounds for Algebraic Computation Trees*, Proc. 15th ACM STOC, 1983, pp. 80-86.
2. L. Babai, B. Just and F. Meyer auf der Heide, *On the Limits of Computations with the Floor Functions, Information and Computation*, 78 (2), 1988, pp. 99-107.
3. J. W. S. Cassels, *An Introduction to the Geometry of Numbers*, Springer, Berlin, 1959; second printing, 1971.
4. D. Dobkin and R. Lipton, *A Lower Bound of $1/2n^2$ on Linear Search Programs for the Knapsack Problem*, J..C.S.S., Vol. 16, 1975, pp. 417-421.

5. J. HASTAD, B. JUST, J. LAGARIAS and C. P. SCHNORR, *Polynomial Time Algorithms for Finding Integer Relations Among Real Numbers*, Proc. STACS, 1986, pp. 105-118.
6. P. KLEIN and F. MEYER AUF DER HEIDE, *A Lower Bound for the Knapsack Problem on Random Access Machines*, Act. Inf., Vol. 19, 1983, pp. 385-395.
7. H. W. LENSTRA Jr, *Integer Programming with a Fixed Number of Variables*, Report 81-03, Mathematisch Instituut, Amsterdam, 1983.
8. J. C. LAGARIAS, H. W. LENSTRA Jr. and C. P. SCHNORR, *Karkine-Zolotareff Bases and Successive Minima of a Latice and its Reciprocal Lattice*, preprint 1986.
9. F. MEYER AUF DER HEIDE, *Lower Bounds for Solving Linear Diophantine equations on Random Access Machines*, J.ACM., Vol. 32 (4), 1985, pp. 929-937.