5. Upper bounds

Objekttyp: **Chapter**

Zeitschrift: **L'Enseignement Mathématique**

Band (Jahr): **27 (1981)**

Heft 1-2: **L'ENSEIGNEMENT MATHÉMATIQUE**

PDF erstellt am: **18.04.2024**

$C_{t-1,s} = b_{t-1}$ and $C_{t-1,s+1} = c_{t-1}$. (Naturally these values must imply the state q_a and the headposition s_t at time t.) Now player A is allowed to doubt one of these three claims, by playing the integer $s' \in \{s-1, s, s+1\}$, and player E has to justify his claim for $C_{t-1,s'}$ by claiming values for $C_{t-2, s'-1}$, $C_{t-2,s'}$ and $C_{t-2,s'+1}$ which imply his value for $C_{t-1,s'}$ etc. Finally the value claimed for $C_{0s''}$ is checked by comparison with the s''-th input symbol. If it is correct, then player E, otherwise player A wins.

If w is accepted by M, then the winning strategy for player E is to make always correct claims. If w is not accepted by M, then player A has a winning strategy. He always doubts one of the wrong claims of player E.

5. Upper bounds

PROPOSITION. 1. *For all* $p \geqslant 0$, *the* $\exists^p\forall\exists^*$ *class is logspace transformable to the monadic* $\exists\,\forall\exists^*$ *class via length order* n.

2. *The* $\exists^*\forall\exists^*$ *class is logspace transformable to the monadic* $\exists^*\forall\exists^*$ *class via length order* $n^2/\log n$.

Proof. The main ideas of this proof are due to Lewis [27, Lemma 7.1] and Ackermann [2, Section VIII.1]. Given a formula F of the class $\exists^p\,\forall\exists^q$ with prefix $\exists x_1 \dots \exists x_p \forall y \exists z_1 \dots \exists z_q$ and matrix M, let S be the set of atomic formulas in M. We define the set S' by $S' = S \cup \{A\,[y/x_i] \mid A \in S$ and $1 \leqslant i \leqslant p\}$.

Let $S' = \{A_1, \dots, A_r\}$.
Then $|S'| = r \leqslant (p+1)\,|S|$.

Now we change the matrix M of F to get the formula F' with matrix M' by replacing (for $j = 1, \dots, r$) all occurrences of the atomic formula A_j by $P_j(y)$ (for a new monadic predicate symbol P_j) and by adding — as a conjunct to M — a set B of biconditionals.

The set B is constructed to ensure that every Herbrand model α' of the functional form of the formula F' (with matrix M') defines immediately a model α of the functional form of F by $|\alpha| = |\alpha'|$,

$c_k^\alpha = c_k^{\alpha'} = c_k$, $k = 1, \dots, p$ (where c_k is the replacement of x_k in the functional forms of F and F'),

$f^\alpha_k = f^{\alpha'}_k$, $k = 1, \ldots, q$ (where $f_k(y)$ is the replacement of z_k in the functional forms of F and F'),

$P^\alpha(b_1, \ldots, b_n) = P^{\alpha'}_j(b)$, if $A_j \in S'$, $b \in |\alpha'|$, $b_1, \ldots, b_n \in |\alpha|$ and there exist variables $v_1, \ldots, v_n$ fulfilling for all i, k the following properties:

a) $A_j = P(v_1, \ldots, v_n)$,

b) if $v_i = x_k$ then $b_i = c^\alpha_k$,

c) if $v_i = y$ then $b_i = b$,

d) if $v_i = z_k$ then $b_i = f^\alpha_k(b)$.

$P^\alpha(b_1, \ldots, b_n)$ is defined arbitrarily (e.g. false) if no such A_j and b exist. There might exist several A_j and b having these properties. To ensure that in this case the definition of $P^\alpha(b_1, \ldots, b_n)$ is correct, i.e. independent of the particular choice of A_j and b, we conjoin the set B of biconditionals to the matrix M.

Take any n-tupel $(b_1, \ldots, b_n) \in |\alpha|^n$. In the following cases, several $A_j \in S'$ and $b \in |\alpha|$ might satisfy the conditions a), b), c), d):

1. $\{b_1, \ldots, b_n\} \subseteq \{c^\alpha_1, \ldots, c^\alpha_p\}$.
2. There is a b' in $\{c^\alpha_1, \ldots, c^\alpha_p\}$ such that $\{b_1. \ldots, b_n\} \subseteq \{c^\alpha_1, \ldots, c^\alpha_p, f^\alpha_1(b'), \ldots, f^\alpha_q(b')\}$.
3. There is a b'' in $\{b_1, \ldots, b_n\}$, such that $\{b_1, \ldots, b_n\} \subseteq \{c^\alpha_1, \ldots, c^\alpha_p, b''\}$.

To make the definition correct in case 1, we add to B the following biconditionals:

If there is an A_j in S' such that $A_j = P(v_1, \ldots, v_n)$ with $\{v_1, \ldots, v_n\} \subseteq \{x_1, \ldots, x_p\}$, we add

$$P_j(y) \leftrightarrow P_j(x_1)$$

If $A_j = P(v_1, \ldots, v_n)$ with $\{v_1, \ldots, v_n\} \subseteq \{x_1, \ldots, x_p, y\}$ and $A_j[y/x_i] = A_{j'}[y/x_k]$ (for $A_j \neq A_{j'}$), then we add

$$P_j(x_i) \leftrightarrow P_{j'}(x_k).$$

Note: Here the length of the monadic formula might grow quadratically in p.

To make the definition correct in the case when 2 but not 3 holds, we add to B for all j, j', i with $A_j[y/x_i] = A_{j'}[y/x_i]$ the formula

$$P_j(x_i) \leftrightarrow P_{j'}(x_i).$$

To make the definition correct, when 3. but not 2. holds, we add to B the following biconditionals.
For all j, j', k such that $A_j = P(v_1, ..., v_n)$ with

$$y \in \{v_1, ..., v_n\} \subseteq \{x_1, ..., x_p, y\}$$

and $A_j [y/z_k] = A_{j'}$, we add

$$P_j(z_k) \leftrightarrow P_{j'}(y)$$

If both 2. and 3. but not 1. hold, and if there are atomic formulas A_j and $A_{j'}$, such that A_j contains y but no variables of $\{z_1, ..., z_q\}$ and $A_j [y/z_k] = A_{j'} [y/x_i]$, we have to make sure that

$$P_j^{\alpha'}(f_k^{\alpha'}(c_i^{\alpha'})) = P_{j'}^{\alpha'}(c_i^{\alpha'}) .$$

But in this case S' contains an $A_{j''}$ with

$$A_{j''} = A_j [y/z_k]$$

and we have added the formulas:

$$P_j(z_k) \leftrightarrow P_{j''}(y) \quad \text{(case 3)}$$

and

$$P_{j''}(x_i) \leftrightarrow P_{j'}(x_i) \quad \text{(case 2)}$$

Hence

$$P_j^{\alpha'}(f_k^{\alpha'}(c_i^{\alpha'})) = P_{j''}^{\alpha'}(c_i^{\alpha'}) = P_{j'}^{\alpha'}(c_i^{\alpha'})$$

It is not obvious that the transformation from formula F to formula F' can be done in logarithmic space, because F might contain variables or predicate symbols with excessively long indices. But then a simple trick solves the problem. Instead of writing such an index on a work tape, only a pointer (= position number) to its location on the input tape is stored on a work tape.

If $|F| = n$, then at most $O(n/\log n)$ different atomic formulas appear in F (i.e. $|S| = O(n/\log n)$). The number $|S'|$ of different atomic formulas in F' is then bounded by $c(p+1)|S|$. Hence the transformation from F to F' is via length order n for constant p and via length order $n^2/\log n$ in general (i.e. for $p = O(n/\log n)$). □

Problem. Is there an efficient transformation from the $\exists^* \forall \exists^*$ class to the monadic $\exists^* \forall \exists^*$ class via length order n ?

THEOREM (Upper bound). *The satisfiability of the monadic prefix class* $\exists^* \forall \exists^*$ *is decidable by an alternating Turing machine* M *in space*

$O(n/\log n)$. Furthermore M enters no universal states for formulas of the subclass $\exists^* \forall \exists$.

Proof. Let the input F be the monadic formula

$$\exists x_1 \dots \exists x_p \forall y \exists z_1 \dots \exists z_q F_0$$

with F_0 quantifier-free. It is easy to find out if the input has this form or not. Let F_0 contain m different atomic formulas. Then $m = O(n/\log n)$ for $n = |F|$.

Let $(v_1, \dots, v_{p+q+1})$ be $(x_1, \dots, x_p, y, z_1, \dots, z_q)$ and let $A_1, \dots, A_m$ be the atomic formulas $P_j(v_i)$ of F_0 in lexicographical order according to (i, j).

$T_1, \dots, T_m$ is a sequence of truth values for the atomic formulas. (The atomic formula A_k is interpreted to be true if T_k = true.)

The alternating Turing machine M executes the following satisfiability test:

Program

1. begin

  for all k such that the atomic formula A_k contains an x_i, choose existentially T_k to be true or false;
  for $r := 1$ to max $(1, p)$ do
  begin

2. for all k, k', j such that A_k is $P_j(y)$ and $A_{k'}$ is $P_j(x_r)$ do $T_k := T_{k'}$;

3. for all k, j such that A_k is $P_j(y)$ and $P_j(x_r)$ does not appear in F do choose existentially a value of {true, false} for T_k;

4. for counter $:= 1$ to 2^m do
  begin

5. for all k such that A_k is a $P_j(z_i)$ do choose existentially a truth value for T_k; check that the interpretation of the atomic formulas A_k $(k = 1, \dots, m)$ by T_k gives the value true to the matrix F_0, otherwise stop rejecting;

7. if $q = 0$ then goto E;
  if $q = 1$ then $s := 1$ (i.e. $z_s = z_1$);
  if $q > 1$ then choose universally a value from $\{1, \dots, q\}$ for s;

8. for all k, k', j such that A_k is $P_j(y)$ and $A_{k'}$ is $P_j(z_s)$ do $T_k := T_{k'}$;

9. for all k such that (for any j) A_k is $P_j(y)$ and $P_j(z_s)$ does not appear in F do choose existentially a truth value for T_k;
end;

E : end;

stop accepting;
end.

To execute this program, the alternating Turing machine M uses only space

m to count to 2^m,
m to store $T_1, \ldots, T_m$,

$\log p < \log m$ to store r,

$c \log n$ for anxillary storage, especially to store position numbers of certain information on the input tape, e.g. long indices, which are not copied to the work tapes.

Because $m = O(n/\log n)$, there is an upper bound $O(n/\log n)$ (independent of p and q) for the space used by M.

We have to show that the above program decides satisfiability of the formula F correctly.

Let $F' = \forall y F_0'$ be the functional form of $F = \exists x_1 \ldots \exists x_p \forall y \exists z_1 \ldots \exists z_q F_0$, obtained by replacing x_i by c_i and z_i by $f_i(y)$.

a) Let F' (and F) be satisfiable and let α be a model of F'.
We think the program of M extended by:

before 2. $b := c_r^\alpha$

before 8. $b := f_s^\alpha(b)$

Then good existential choices for the truth values T_k are

if $A_k = P_j(x_i)$ then $T_k := P_j^\alpha(c_i^\alpha)$
if $A_k = P_j(y)$ then $T_k := P_j^\alpha(b)$
if $A_k = P_j(z_i)$ then $T_k := P_j^\alpha(f_i^\alpha(b))$

The computation tree defined by these existential choices accepts the formula F.

b) Assume the alternating Turing machine M accepts the formula F. Then each minimal accepting computation tree (without unnecessary branches) of M with input F can be used to construct a Herbrand model α of F'.

Note that the Herbrand universe

$$|\alpha| = \{c_1, ..., c_p, f_1(c_1), ..., f_2(f_1(c_3)), ...\}$$

(as a set of terms) and the functions $f_1^\alpha, ..., f_q^\alpha$ of a possible Herbrand model of F' are uniquely defined. We have to define the predicates $P_1^\alpha, P_2^\alpha, ...$.

We look at the program extended by

$$b := c_r^\alpha \qquad \text{(before 2) and}$$
$$b := f_s^\alpha(b) \qquad \text{(before 8) as in a).}$$

All elements of $|\alpha|$ with nesting depth $\leqslant 2^m$ are assigned to b somewhere in the accepting computation tree. The current values of the sequence $T_1, ..., T_m$ define some truth values of predicates in $c_1^\alpha, ..., c_p^\alpha, b, f_1^\alpha(b), ..., f_q^\alpha(b)$ by

$$\begin{aligned} P_i^\alpha(c_k^\alpha) &= T_j \quad \text{if} \quad A_j = P_i(x_k) \\ P_i^\alpha(b) &= T_j \quad \text{if} \quad A_j = P_i(y) \\ P_i^\alpha(f_k^\alpha(b)) &= T_j \quad \text{if} \quad A_j = P_i(z_k)\,. \end{aligned}$$

The other truth values of the predicates P_i^α are defined arbitrarily. This method of defining predicates of b is used on each path in the tree $(|\alpha|, f_1^\alpha, ..., f_q^\alpha)$, only until the first repetition of all truth values on that path. That happens on each path in a depth $\leqslant 2^m$. Let b' be the node on the path to b with the same truth values for all predicates as b. Then (inductively) the predicates are defined to have the same values on the subtree with root b as on the subtree with root b'. The so constructed structure α is a model of F. □

COROLLARY 1 (Lewis [27]). *The set of satisfiable formulas of the monadic* $\exists^* \forall \exists^*$ *class is (for a constant* $c > 1$*) in* $DTIME(c^{n/\log n})$.

Proof. The alternating Turing machine of the upper bound theorem can be simulated in deterministic time $c^{n/\log n}$. □

The direct construction of a deterministic $c^{n/\log n}$ time decision procedure of Lewis [27] is easier. He starts with a big structure (with 2^m elements, where m is the number of predicate symbols), and eliminates bad elements of this structure, to get either a model or the non-existence of a model.

We have chosen the decision procedure by an alternating Turing machine to get the following result for free.

COROLLARY 2. *The satisfiable formulas of the monadic* $\exists^* \forall \exists$ *class are in* $NSPACE\,(n/\log n)$.

Proof. The universal states of the alternating Turing machine M which decides the monadic $\exists^* \forall \exists^*$ class are not used for the subclass $\exists^* \forall \exists$. If we drop them, we get a nondeterministic Turing machine. □

By combining the proposition with the upper bound theorem we get immediately.

COROLLARY 3. *The satisfiable formulas of the* $\exists^* \forall \exists^*$ *class are in* $DTIME\,(c^{(n/\log n)^2})$ *for some* c. □

COROLLARY 4. *The satisfiable formulas of the* $\exists^* \forall \exists$ *class are in* $NSPACE\,((n/\log n)^2)$. □

Lewis [27] claims the same time bound in Corollary 3 as for the monadic case. But this seems not to work. For example, if $P(x_1, y), \ldots, P(x_p, y)$ and $P(y, x_1), \ldots, P(y, x_p)$ appear in the formula, then p^2 truth values for $P^\alpha(c_i^\alpha, c_j^\alpha)$ $(i, j = 1, \ldots, p)$ have to be guessed.

But these upper bounds are not very good, as e.g. in Corollary 3 the Turing machine could be replaced by one which works a short time ($O((n/\log n)^2)$ steps) nondeterministically and then only $c^{n/\log n}$ steps deterministically.

The $\exists^* \forall$ *class*

Formulas of the $\exists^* \forall$ class are transformed by our procedure in monadic formulas again of the $\exists^* \forall$ class. For these formulas, the procedure of the upper bound theorem works in nondeterministic polynomial time. On the other hand the $\exists^* \forall$ class is certainly more difficult than propositional calculus. Therefore the set of satisfiable formulas of the $\exists^* \forall$ class is *NP*-complete. (*NP*-completeness is discussed in [15].)

In fact, as the Herbrand models of the satisfiable formulas of the $\exists^p \forall^q$ class, have only max $(p, 1)$ elements, it is easy to see that the satisfiability problem for all the following classes in *NP*-complete:

a) $\exists^p \forall^q \quad p + q \geqslant 1$

b) $\exists^* \forall^q \quad q \geqslant 0$

c) $\forall^*$

d) $\exists \forall^*$

But the classes $\exists\exists\forall^*$ and $\exists^* \forall^*$ need $NTIME\; c^{n/\log n}$ resp. c^n.