A NOTE ON SOME LANGUAGES IN UNIFORM *ACC*

David A. Mix Barrington
James Corbett

Computer and Information Science Department
University of Massachusetts

COINS Technical Report 89-74

A note on some languages in uniform ACC

David A. Mix Barrington[1]
James Corbett[2]
COINS Dept., U. of Massachusetts
Amherst, MA 01003, U.S.A.
July 25, 1989

1. Abstract

ACC is the class of languages recognized by circuit families with polynomial size, constant depth, and unbounded fan-in, where gates may calculate the AND, OR, or MOD c function for constant c. Robust uniformity definitions for ACC and related classes were given by Barrington, Immerman, and Straubing [3]. Here we show that uniform ACC contains all semi-linear or rational sets of integer vectors, using binary notation (sharpening a result of Ibarra, Jiang, Ravikumar, and Chang [10]).

2. Introduction

Recent work of Barrington, Immerman, and Straubing has given robust definitions of uniformity for various circuit complexity classes within NC^1(languages recognizable by circuit families of fan-in two and depth $O(\log n)$— see Cook [6] for an overview of circuit complexity theory). These classes are defined as those languages recognizable by circuit families with polynomial size, constant depth, and unbounded fan-in, with a variety of possible functions computed by individual gates. If these functions are the usual AND and OR functions (where inputs to the circuit are input variables or their negations), we have the class AC^0. If gates are allowed which add their input modulo some constant (i.e. a gate outputs a 1 if and only if the number of its inputs which are one is divisible by c), we have the class ACC. The constant must be fixed for the entire circuit family. It is known that AC^0 is different from ACC [1,8] and limits are known in the subcase where the constant c

[1]Former name David A. Barrington. Supported by NSF Computer and Computation Theory grant CCR-8714714.
[2]Supported by NSF grant CCR-8812567.

is prime [15,17]. If the gates can compute threshold functions, we get the class TC^0 [5,9,13]. With certain gate functions defined in terms of any non-solvable group, we get all of NC^1 [2].

The uniformity notion we will use is *log-time* uniformity, originally defined by Buss [4]. This requires that a deterministic Turing machine can answer simple questions about the n^{th} circuit in the family in $O(\log\ n)$ time. More precisely, let the *direct connection language* of a circuit family be the set of all tuples $< t, a, b, 0^n >$ where a and b are numbers of nodes in the n^{th} circuit, b is a child of a, and node a is of type t (e.g. AND, OR). Then we say that a circuit family is log-time uniform if and only if its direct connection language can be recognized in deterministic log time. We will use the characterization of log-time uniform $AC^0$and ACC from [3] as those languages for which membership can be *expressed* by first-order formulas (see [11] for an overview of logical expressibility as a complexity measure).

In such formulas, variables range over positions in the input (from 1 to n) which is read with the special predicate $\pi_a(i) \equiv$"the i^{th} input character is an a". Variables can also be compared. For example, the regular language 0^*1^+ can be expressed as

$$\exists x \forall y[((y < x) \rightarrow \pi_0(y)) \wedge ((y \geq x) \rightarrow \pi_1(y))]$$

We can optionally add the special predicate $BIT(i,j) \equiv$"the i^{th} bit of the binary representation of j is a 1". We can also add special *modular counting quantifiers* $Q_{m,a}$ for any positive integer m and any integer a such that $0 \leq a < m$. If $\psi(x)$ is a sentence with one free variable, then the formula $Q_{m,a}x : \psi(x)$ is defined to be true if and only if the number of x's for which $\psi(x)$ is true is congruent to a mod m. For example, the language of binary strings with an odd number of ones can be expressed as $Q_{2,1}x : \pi_1(x)$. It has been shown [3] that the class $FO + BIT$ (languages expressible by first-order formulas using the BIT predicate) is exactly log-time uniform AC^0, and that the class $FOC + BIT$ (languages expressible by first-order formulas with modular counting quantifiers and BIT) is exactly log-time uniform ACC. Without BIT, the classes FO and FOC are interesting subclasses of the regular languages, with FO being the star-free regular languages [12,18].

ACC is a very limited complexity class, but we do not know how to separate it from NC^1, or even from a much larger class like NP. It is useful to have examples of languages known to be in these various complexity classes, to get a better idea of how to place limits on their power. In this note, we examine one class of languages, binary encodings of semi-linear sets, showing them to be within ACC.

3. Semi-linear sets in ACC

A *linear set* is the nonegative span of a finite set of vectors offset by a single vector. More formally, the linear set corresponding to the vectors $\mathbf{a}, \mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m \in N^k$, is

$$\{\mathbf{u} | \mathbf{u} = \mathbf{a} + r_1\mathbf{b}_1 + r_2\mathbf{b}_2 + \ldots + r_m\mathbf{b}_m, r_i \in N\}$$

A *semi-linear set* is a finite union of linear sets. In addition, a *simple* set is a linear set in which the vectors b_i are linearly independent. A *semi-simple* set is a finite union of simple sets.

The binary encodings of vectors in a fixed semi-linear set forms a language. Ibarra, Jiang, Ravikumar, and Chang assert that these langauges are in NC^1 [10]. We improve this result by showing:

Theorem 1 *Binary encodings of semi-linear sets are in log-time uniform ACC.*

Proof: Since a semi-linear set is the finite union of linear sets, we can simply test the input vector for membership in all these linear sets in parallel and then take the disjunction of the results. Thus we can restrict our attention to determining membership in linear sets. Given $\mathbf{u}$, the binary encoding of a k dimensional vector, we want to determine if there exist nonegative integers $r_1, \ldots, r_m$ such that $\mathbf{u} = \mathbf{a} + r_1\mathbf{b}_1 + r_2\mathbf{b}_2 + \ldots + r_m\mathbf{b}_m$. Using linear algebra, we can state this as the problem of finding a nonegative integer solution to the system $B\mathbf{r} = \mathbf{u} - \mathbf{a}$ where B is a matrix whose i^{th} column is $\mathbf{b}_i$. There are two cases. If the linear set is simple, then there is a unique solution over the rationals given by Cramer's rule:

$$r_i = \frac{\|B_i\|}{\|B\|}$$

where B_i is the matrix obtained by replacing the i^{th} column of B with $\mathbf{u} - \mathbf{a}$, and $\|A\|$ is the determinant of matrix A. We must only check to make sure the rational solution is both nonegative and integer.

Note that since $\mathbf{a}$ and B are constants, $\|B_i\|$ is a linear expression in the input $\mathbf{u}$. Since we can add integers and multiply them by constants in $FO + BIT$ [11], we can compute the numerator in Cramer's rule and from it determine the sign of the solution. To verify that the solution is an integer vector, we must test this numerator for divisibility by the constant $c = \|B\|$.

This can be done by a simple automaton which scans across the binary digits of the number from left to right and keeps track of the remainder, as in short

division. Let $w \in (0+1)^*, b \in \{0,1\}$ and $\phi(w)$ be the remainder when the binary number w is divided by c. Then $\phi(wb) = (2\phi(w) + b) \bmod c$. Since the transition function implements a linear map, it is easy to verify that the syntactic monoid of the automaton is solvable (a monoid is solvable if and only if any group contained in the monoid is solvable) and hence its output is computable in FOC by [18].

Since all this can be done in $FOC + BIT$, we are in log-time uniform ACC by the results of [3].

The second case is when the linear set is not simple. This case reduces to the first by a result of Eilenberg and Schützenberger [7]. They showed that all linear sets are semi-simple, and hence that a nonsimple linear set can be decomposed into a finite union of simple sets. □

This result is optimal in the sense that there are semi-linear sets known not to be in AC^0. For example, consider the simple set L_3 of all binary encodings of integers divisible by 3. We will show L_3 is not even in non-uniform AC^0. By well known circuit complexity results [1,8] we know that computing the MOD-p function, i.e. determining whether the number of ones in a binary string is congruent to zero modulo a prime p, is not in non-uniform AC^0. If L_3 were in AC^0, then we could compute MOD-p in AC^0 as follows. Observe that if $x = \sum x_i 2^i$ then since $2^i \equiv 1 \pmod 3$ for even i and $2^i \equiv 2 \pmod 3$ for odd i,

$$x \equiv \sum_k x_{2k} + 2\sum_k x_{2k+1} \pmod 3$$

Thus given a binary string y, we could construct in constant depth the binary string z by inserting 0's between every two bits of y: $z_{2i} = y_i, z_{2i+1} = 0$. Since z (interpreted as a binary number) is divisible by 3 if and only if the number of 1's in y is congruent to zero mod 3, we could compute the MOD-3 function in AC^0 using a hypothetical AC^0 circuit for L_3. This being impossible, we can conclude that no family of AC^0 circuits exists for L_3.

Essentially, the proof of Theorem 1 shows that it is possible to determine in ACC whether or not an integer solution $\mathbf{r}$ exists to the linear system $B\mathbf{r} = \mathbf{x}$ where x is input and B is a constant matrix. This allows us to solve a limited case of the integer linear programming problem where the number of constraints and the coefficients of the variables in those constraints are restricted to be constant. The inequality constraints of a linear programming problem can be transformed into equality constraints by standard methods and a nonegative solution found as above.

As a final note, we observe that this technique also shows that all rational subsets of Z^k are in uniform ACC. The *rational subsets* of Z^k are the smallest class

of subsets containing the finite sets and closed under finite union, sum, and star [14] where sum and star are defined as follows

$$\begin{aligned} X+Y &= \{x+y | x \in X,\ y \in Y\} \\ X^* &= \text{subspace of } Z^k \text{ generated by } X \end{aligned}$$

By [7], every rational set is semi-simple, and hence in uniform ACC by the techniques presented here.

References

[1] M. Ajtai, "Σ_1^1 formulae on finite structures," *Annals of Pure and Applied Logic* **24** (1983), 1-48.

[2] D. A. Barrington, "Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1," *J. Comp. Syst. Sci.* **38:1** (1989), 150-164.

[3] D. A. M. Barrington, N. Immerman, and H. Straubing, "On uniformity within NC^1," *Structure in Complexity Theory: Third Annual Conference*(Washington: IEEE Computer Society Press, 1988), 47-59. Revised version *J. Comp. Syst. Sci.*, to appear. Available as University of Massachusetts COINS Technical Report 88-60.

[4] S. R. Buss, 'The Boolean formula value problem is in ALOGTIME," *18th ACM STOC Symp.* (1987), 123-131.

[5] A. K. Chandra, L. J. Stockmeyer, and U. Vishkin, "Constant depth reducibility," *SIAM J. Comp.* **13:2** (1984), 423-439.

[6] S. A. Cook, "A taxonomy of problems with fast parallel algorithms," *Information and Control* **64** (1985), 2-22.

[7] S. Eilenberg and M. P. Schützenberger, "Rational sets in commutative monoids," *Journal of Algebra* **13** (1969), 173-191.

[8] M. Furst, J. B. Saxe, and M. Sipser, "Parity, circuits, and the polynomial-time hierarchy," *Math. Syst. Theory* **17** (1984), 13-27.

[9] A. Hajnal, W. Maass, P. Pudlák, M. Szegedy, and G. Turán, "Threshold circuits of bounded depth," *28th IEEE FOCS Symp.* (1987), 99-110.

[10] O. H. Ibarra, T. Jiang, B. Ravikumar, and J. H. Chang, 'On some languages in NC^1," *VLSI Algorithms and Architectures: Proc. of 3rd Aegean Workshop on Computing, 1988.*

[11] N. Immerman, "Expressibility and parallel complexity," *SIAM J. Comp.* **18:3** (1989), 625-638.

[12] R. McNaughton and S. Papert, *Counter-Free Automata* (Cambridge, Mass.: MIT Press, 1971).

[13] I. Parberry and G. Schnitger, "Parallel computation with threshold functions," *J. Comp. Syst. Sci.* **36:3** (1988), 278-302.

[14] P. Péladeau, "Logically defined subsets of N^k," *Proc. 14th annual Symp. Mathmetical Foundations of Comp. Sci.*, Porabka Kozubnik, Poland (1989), to appear.

[15] A. A. Razborov, "Lower bounds for the size of circuits of bounded depth with basis $\{\&, \oplus\}$," *Mathematicheskie Zametki* **41:4** (April 1987), 598-607 (in Russian). English translation *Math. Notes Acad. Sci. USSR* **41:4** (Sept. 1987), 333-338.

[16] W. L. Ruzzo, "On uniform circuit complexity," *J. Comp. Syst. Sci.* **21:2** (1981), 365-383.

[17] R. Smolensky, "Algebraic methods in the theory of lower bounds for Boolean circuit complexity," *19th ACM STOC Symp.* (1987), 77-82.

[18] H. Straubing, D. Thérien, and W. Thomas, "Regular languages defined with generalized quantifiers," *Proc. 15th ICALP*, Springer Lecture Notes in Computer Science **317** (1988), 561-575.