A Clarification Concerning the #L Hierarchy

Eric Allender*
Department of Computer Science, Rutgers University
P.O. Box 1179, Piscataway, NJ 08855-1179,USA
e-mail: allender@cs.rutgers.edu

October 29, 1997

Abstract

In [AO96], it is stated (without proof) that if $\mathrm{NC}^1(\#\mathrm{L}) = \mathrm{AC}^0(\#\mathrm{L})$*, then the #L hierarchy collapses to some level. This note provides a proof of that claim. The reader is referred to [AO96] for all background, definitions, and motivation. All of the circuit complexity classes mentioned in this note are logtime-uniform [BIS90], unless otherwise specified.*

1 The Proof

The proof consists of establishing the following fact.

Fact 1.1 *The set of languages* $NC^1(DET)$ *has a complete set under logspace many-one reducibility.*

This fact is sufficient to establish the claim made in [AO96]. To see this, assume that $\mathrm{NC}^1(\mathrm{DET}) = \mathrm{AC}^0(\mathrm{DET})$. By Claim 1.1, there is a complete set for $\mathrm{AC}^0(\mathrm{DET})$. Since $\mathrm{AC}^0(\mathrm{DET})$ is equal to $\mathrm{L}^{\#\mathrm{L}^{\#\mathrm{L}^{\cdots^{\#\mathrm{L}}}}}$, this complete set is in some fixed level of this hierarchy, and thus $\mathrm{AC}^0(\mathrm{DET})$ collapses to this level.

It remains for us to establish Fact 1.1.

*Supported in part by NSF grant CCR-9509603.

First, let's come up with a definition of a "canonical" way for a log-time machine to specify a circuit. Let M be a clocked[1] 3-tape[2] Turing machine running for $c \log n$ time for some c.

Let us say that M is *k-good on length n* if for all p of length $\leq k \log n$, $M(n,p)$ is a string in the set

$$\{(b, \text{ORACLE}), (2, \text{AND}), (2, \text{OR}), (1, \text{NOT}), (i, \text{INPUT})\}$$

where $|b| + |p| \leq k \log n$, and $i \leq n$, and such that if $|p| = k \log n$, then $M(n,p) = (i, \text{INPUT})$ for some i, and for each prefix q of p, if $M(n,q) = (j, \text{INPUT})$, then $i = j$. (Intuitively, p is an encoding of a path from the output gate to a gate g in the NC^1 circuit, and M, on input (n,p), is producing as output the fan-in of the g, and the type of gate that g is. The other conditions are merely for technical convenience. The depth of the circuit is $k \log n$.)

Note that – for a suitable encoding of clocked Turing machines – the following language is in uniform AC^0, for each c and k: $\{M, n : M$ is a clocked Turing machine running in $c \log n$ time, and M is k-good on length $n\}$. (This does depend somewhat on the encoding of Turing machines. However, note that for any reasonable encoding of Turings machines M, the language $\{1^M 0^n 1^p 0^i b$: the i-th output bit of $M(n,p)$ is $b\}$ is in Dlogtime-uniform AC^0 (since it is in Dlogtime). Checking if a circuit is 2-good can be expressed as a first-order sentence over a uniform AC^0 predicate, and thus it is in Dlogtime-uniform AC^0.)

Define the circuit $C_{M,c,k,n}$ as follows: If M is a clocked Turing machine rnning in time $c \log n$ and M is k-good on length n, then this is the circuit with gates having labels of the form p, where the type and fan-in of gate p is given by $M(n,p)$. If gate p has fan-in b, then for all strings q of length $|b|$ that lexicographically precede b, the gates that are input to p are the gates pq. (If M is not a clocked Turing machine that is k-good on length n, then the circuit is a circuit that trivially accepts the empty set.)

I claim that the set $C = \{M, x : M$ is a clocked Turing machine running in $5 \log n$ time, and M is 2-good on length $|x|$, and the circuit $C_{M,c,k,|x|}$ accepts x (where the oracle gates give the middle bit[3] of the function DET applied to their inputs), and $|M| \leq \log \log |x|\}$ is complete for $\text{NC}^1(\#\text{L})$.

We need to show that C is in $\text{NC}^1(\#\text{L})$, and that it is hard. Neither seems completely trivial.

---

[1] A clocked TM running in time $c \log n$ is a machine that, on input (n,p)

1. computes $c \log n$ (which is just c times $|n|$).
2. starts a counter that will allow it to execute only $c \log n$ steps.

(Actually, steps (1) and (2) can be begun simultaneously; there are a number of programming tricks with Turing machines that one can use. The point is, (a) there is some purely syntactic part of the Turing machine description that we will call the "clock", (b) this "clock" enforces a run-time on the Turing machine, and (c) every Turing machine is equivalent to a "clocked" Turing machine of comparable complexity.)

[2] Note that Dlogtime-uniform AC^0 and NC^1 have circuits that are Dlogtime-uniform even with this additional restriction that the uniformity machine have three tapes [BIS90].

[3] Any bit of the DETERMINANT can be reduced to the middle bit. (If I want the lower-order bit of f(x), this is the middle-bit of some function GapL function g(x) defined as $f(x)2^{n^k}$ for some k.)

First, let's show that it's in $\mathrm{NC}^1(\#\mathrm{L})$. Let m be fixed. We'll describe the circuit accepting C for inputs of length m. First, given input M, x, where $|x| = n$, the circuit will evaluate the AC^0 predicate to check that M runs for $5 \log n$ time and is 2-good on length $|x|$; thus let's assume that M is good, and let's concentrate on length n. For each string p, there will be circuitry evaluating $M(n, p)$. The output of the circuit our circuit accepting C, of course, is the value of gate λ in circuit $C_{M,c,k,n}$ on input x. (Recall that λ (the empty string) is the name of the output gate of $C_{M,c,k,n}$.) Here is the circuitry that will evaluate any given gate p. With NC^0 circuitry, (with "free" calls to the AC^0 predicates that are computed only once, given $M(n, p)$) we can compute the "type" of gate p. If the gate is of type INPUT, then a subcircuit of depth $\log n$ can compute the input bit i to which gate p is connected. If the gate is of any other type, then a subcircuit of depth $\log b$ can compute the fan-in b of gate p. (That is, near the "top" of this circuit, there will be gates checking if the fan-in is 2, and attempting to compute the AND of the inputs of gate p; near the bottom of the circuit, there will be gates checking if the fan-in is n^4, and attempting to compute DET(the values of the input gates pq), etc.)[4] The total circuit depth required to evaluate a gate of fan-in d is $O(\log d)+$ depth of its inputs. This is all that's required in order to show that this is in $\mathrm{NC}^1(\#\mathrm{L})$.

Now, let's show hardness.

For this, we need to show that anything accepted by $\mathrm{NC}^1(\#\mathrm{L})$ circuits is accepted by circuits of the form $C_{M,c,k,n}$ for some k-good Turing machine M, for some c and k. (If we have this, then a standard "padding" reduction will show that our set C is complete.) What we need is that a log-time machine, given a path p, can compute the type and fan-in of the gate that is reached by following path p from the output gate. (By "following a path p", I mean that at each oracle gate of fan-in b, $\log b$ bits are used to determine the input wire from the oracle gate that is followed.)

Let A be accepted by *logspace-uniform* $\mathrm{NC}^1(\#\mathrm{L})$ circuits C_n. Note that there is a *very* uniform $\mathrm{NC}^1(\#\mathrm{L})$ family of circuits recognizing the language $\{(n, p, t, i)$: g is the gate reached by following path p in C_n, and either i is 0 and the gate has type t, or $i \leq logb$ and t is the i-th bit of the binary representation of the fan-in of gate g.$\}$ That is, the $\#L$ oracle can be used to determine the name

[4] Actually specifying the labeling M uses will be a bit messy. The circuit has a *very* regular structure:

- Use OR gates to guess the type of the gate g.
- Use ANDs to
  - check that the guess is correct, and
  - simulate the gate.

  The circuits of type (1) are all similar. To do part (2), we (a) Use OR gates to guess the next bit of the fan-in b of g (b) use AND gates to check that this bit is correct Once we have the entire fan-in, we have a gate that actually simulates the gate of the original circuit, and then we repeat the process for the inputs to gate g.

Looking at a path name, it is not too hard to compute what type of gate one is at, and what the fan-in needs to be. However, it will be a bit messy to describe.

of the gate reached by following a given path, and to determine the output of the uniformity machine for C_n. Now it is not hard to use oracle gates of this form to build a new circuit family recognizing A, and having the property that the circuits are specified by a k-good machine M. The details are left to the interested reader.

This completes the sketch of the proof.

The language C we build is complete under projections (which is more restrictive than being complete under many-one reductions).

The same construction can be carried out for other functions f in place of DET. However, if f is not at least hard for NC^1 under projections, then the language C we build won't necessarily be hard for $\mathrm{NC}^1(f)$. (Furthermore, if we want to simulate logspace-uniform $\mathrm{NC}^1(f)$, then this construction will require that f be hard for logspace.)

Acknowledgment: Thanks to Meena Mahajan and V. Vinay for pointing out that the claim made in [AO96] requires proof.

References

[AO96] Allender and M. Ogihara. Relationships among PL, #L, and the determinant. *RAIRO - Theoretical Information and Application*, **30** (1996), 1–21.

[BIS90] D. A. Mix Barrington, N. Immerman, and H. Straubing. On uniformity within NC^1. *Journal of Computer and System Sciences*, 41:274–306, 1990.