The Computational Complexity Column

Eric Allender[1]

Rutgers University, Department of Computer Science
Piscataway, NJ 08855-8019 USA
allender@cs.rutgers.edu

A few short days ago, I received news that a guest column encountered unexpected delays, so this edition of the Computational Complexity Column consists of some hurriedly-compiled

News from the Isomorphism Front[2]

1 The Interview Begins: Questions are Raised about Factorization

Q: In an earlier column in this series [Al97], you gave a very persuasive and highly readable account, arguing that the study of *complexity classes* offers the best hope for understanding the computational complexity of important computational problems.

A: You are too kind.

Q: Yes, I *was* being too kind. I'll try to make up for it now. Honestly, I don't think that the theory of complexity classes has *anything* to say about some of my favorite computational problems. Consider the list:

- Factorization
- Graph Isomorphism
- Graph Automorphism
- Primality Testing
- Greatest Common Divisor
- Perfect Matching

[1]Supported in part by NSF grant CCR-9734918.

[2]Profuse apologies to both Yuri Gurevich, and to Dave Barry, both of whom make much better use of this "question and answer" format than I can hope to.

A: I see your point. None of these problems is known to be complete for any complexity class. But this list simply supports my point: the very fact that this list is so short is one of the unexpected triumphs of complexity theory.

Q: But it is *not* short! There are infinitely many problems that are not complete for any reasonable complexity class. Ladner's construction [La75] shows how to construct such sets.

A: True, but the "computational problems" given by Ladner's lovely construction are very artificial. Empirically, the *overwhelming majority* of the computational problems that people actually want to solve can be shown fairly easily to be complete for one of a small number of natural complexity classes, even under very restrictive notions of reducibility, such as $\leq_m^{AC^0}$ reductions.

Q: Let's get back to my list of problems. Certainly there are some very important problems on this list. And it seems that computational complexity can't say *anything* about *any* of these problems. For instance, are any of these problems even *hard* for any complexity class?

A: This is a good question. First, let's remind the reader what is meant by being "hard" for a complexity class. A is *hard for* $\mathcal{C}$ (where $\mathcal{C}$ is a set of languages) if for every $B \in \mathcal{C}$, there is a function f computed by AC^0 circuits, such that $x \in B$ if and only if $f(x) \in A$. This is a complexity-theoretic notion of a lower bound on the complexity of A. Showing that A is in some complexity class provides a notion of upper bound. If A is complete for $\mathcal{C}$, then these upper and lower bounds match.

Now back to your question. We do know something about the Graph Isomorphism problem. It was recently shown that Graph Isomorphism is hard for NLOG [To98]. (As regards upper bounds, Graph Isomorphism is in NP $\cap$ co-AM $\cap$ LWPP – and I'm probably leaving some classes out of this list. For more about Graph Isomorphism, consult [KST93].)

The situation is quite a bit better for the perfect matching problem. It has been known since [KUW86] that matching is hard for NLOG, and recently an upper bound was presented that in some ways is fairly "close" to the lower bound [AR98] (although I don't want to make this precise here because it would involve a digression about notions of circuit uniformity, etc).

However, for the rest of the problems on your list, I'm afraid that there aren't any theorems regarding hardness or completeness for any of these problems. ... At least not yet ...

Q: Are you hinting that you think someone *might* prove that, say, factorization is complete for some natural complexity class?

A: Absolutely not! In fact, I think that factorization is *not* complete for any such class.

Q: Why not?

A: Well ..., it doesn't have the right "shape". It's structure just isn't right. It doesn't allow the type of coding that you need in order to build a reduction.

Q: Those are some of the most pathetic, *least* satisfactory answers I can imagine! Frankly, I'm quite disappointed. Is that the best you can come up with?

A: *Sigh* ... I'm afraid the best I can do is to present some plausible conjectures. Proving any one of these conjectures would prove that factorization is not complete for any "reasonable" complexity class. Although I can't prove any of these conjectures right now, I think that they might be tractable, and I also hope that they'll make for interesting reading. In fact, **presenting these conjectures is the entire point of this article**. First, however, we will need to make a digression, while we discuss some recent progress on the isomorphism conjecture.

Q: I'm always interested in hearing about progress!

2 Progress on the Isomorphism Conjecture

A: The Berman-Hartmanis Conjecture [BH77] states that all NP-complete sets are isomorphic under bijections computable and invertible in polynomial time. (Here, an isomorphism is nothing more than a bijection; there is no other "structure" that is being preserved. However, this is an abstract way of capturing the intuitive notion that two different "encodings" of a set are still really the "same" set in some sense. For instance, the SAT problem (the set of satisfiable Boolean formulae) can be encoded using round parentheses or square brackets, or with variables $\{x_1, x_2, \ldots\}$ or $\{\mathrm{P}_1, \mathrm{P}_2, \ldots\}$. Such encoding choices are trivial and unimportant from the standpoint of complexity theory. One way to make this precise is to group "equivalent encodings" of a problem into equivalence classes, where an "encoding" can be any "easily-computable" bijection on $\{0,1\}^*$.

Q: I see. So you get one equivalence class for SAT, another class for the Clique problem, one for 3-colorability, and so on.

A: Right – except that all the problems you mentioned are in the *same* class.

Q: Really? Clique is an encoding of SAT?

A: Yes. Berman and Hartmanis [BH77] gave some simple sufficient conditions for two sets to be isomorphic, and with these conditions, it is easy to see that all of the NP-complete problems in [GJ79] are isomorphic.

Q: What about smaller complexity classes, such as P and NLOG?

A: Hartmanis [Ha78] studied analogous conjectures for these classes, using logspace reducibility (and logspace isomorphisms), instead of poly-time. Once again, all of the standard complete sets are easily seen to be isomorphic. Of course, logspace reductions are too powerful to investigate the many interesting subclasses of DLOG (such as ACC^0, TC^0, and NC^1). For these classes, AC^0 reducibility is the most natural notion of reducibility to use (see [Ba89, Bu87], for example), and once again the "standard" complete sets are all seen to be isomorphic [ABI93].[3]

Q: So, I guess that, with so much written about the isomorphism conjectures, most complexity theoreticians believe that the conjectures are true?

A: Far from it. In fact, probably most complexity theoreticians believe that there is a one-way function f (one that is easy to compute but very hard to invert) such that $f(\mathrm{SAT})$ is *not* isomorphic to SAT. Some really interesting work in this direction has been done by [JY85, KLD86, KMR95]...

Q: Let me interrupt. You'll never get this article done on time if you are going to survey the entire field. Are there any surveys you can suggest?

A: Yes. Two very nice ones that spring to mind are [KMR90] and [Yo90]. Some exciting developments (such as [FFK96, JPY94, Ro95]) are too recent to be mentioned in the surveys.

Q: What does this have to do with factorization?

A: I'm getting to that. In spite of the general feeling that the original Berman-Hartmanis conjecture is probably false, it was shown in [AAR96] that the Berman-Hartmanis conjecture for AC^0 reductions is actually *true*.

Q: Please state this more clearly.

A: For any "reasonable" complexity class, all of the sets that are complete under AC^0 reductions are isomorphic under bijections computable and invertible by AC^0 circuits. (In fact, they are computable and invertible by *depth three* AC^0 circuits!)

Q: Is depth three optimal?

A: Yes [AAIPR97].

Q: What is a "reasonable" complexity class?

A: Any class that is closed under many-one reductions computable in TC^0. In particular, NP, P, NC^1, DLOG, BPP, and just about any other complexity class you've ever heard of is "reasonable" in this sense (except for very small classes, such as AC^0 and ACC^0).

Q: So, in light of your earlier comments about $f(\mathrm{SAT})$, when f is a one-way function, does this mean that there are no one-way functions computable in AC^0?

A: Quite the contrary. It was shown in [Ni92] that there are functions f computable in AC^0 that are *very* one-way with respect to depth-three AC^0. Nonetheless, even for such a "random-looking" function f, $f(\mathrm{SAT})$ is just a "trivial" re-encoding of SAT (i.e., it is depth-three AC^0-isomorphic to SAT).

Q: Are there some annoying details about circuit uniformity conditions that you're ignoring in this article?

A: Yes.

Q: This seems to be quite a strong result! Complexity classes are the fundamental objects of study in complexity theory, because the natural computational problems

[3] Actually [ABI93] proves something much stronger, but that would require another digression.

that we are interested in tend to cluster into classes of complete sets for complexity classes. The result of [AAR96] shows that, not only are the complete sets interreducible, but in fact they are isomorphic in a *very* restrictive sense.

A: Remember, this is only true for sets that are complete under AC^0 reducibility.

Q: Yes, but you said that all of the sets that are complete under other reducibilities are also complete under AC^0 reducibilities.

A: If that were true, then we'd have a proof that $P \neq NP$! I merely said that all of the "natural" examples of complete sets (for instance, all of the complete sets considered in [GJ79, GHR95]) are complete under AC^0 reducibilities. Actually, it is shown in [AAIPR97] that there are sets complete for NP under poly-time reducibility, but *not* under AC^0 reducibility.

Q: What does this set look like?

A: It is an encoding of SAT, using a certain kind of error-correcting code. This set (let's call it SAT′) is clearly poly-time isomorphic to SAT, but it is not NP-complete under AC^0 reductions. In fact, there are regular sets (such as the PARITY language) that are not reducible to SAT′ under AC^0 reductions.

Q: Isn't this a counterexample to your claim that all "natural" NP-complete sets are complete under AC^0 reducibility?

A: This depends on your definition of "natural". I think that not many people would say that this is the "natural" way to encode SAT.

Q: We'll let the readers of this column decide what they think of this semantic hair-splitting. But let's get back to the factorization problem.

A: Thank you!

3 Factoring and Cylinders

A: First let's be very precise about what we mean by "the factorization problem". Let us define it to be the following language:

$$\mathrm{FACT} = \{(x, i, b) : \text{the } i\text{th bit of the prime factorization of } x \text{ is } b\}$$

(where the prime factorization is presented as $p_1^{e_1}, \ldots, p_k^{e_k}$, where each exponent $e_i > 0$, and $p_i < p_{i+1}$, so that each number has a unique prime factorization.

Q: What are the complexity-theoretic "upper and lower bounds" on the complexity of FACT?

A: FACT is in NP$\cap$coNP (and in fact it is even in UP$\cap$coUP [Fe92]). Thus FACT is not NP-complete unless NP = coNP = UP = coUP. On the other hand, FACT is not known to be hard for *any* reasonable complexity class under AC^0 reducibility. It is not even known to be hard for $AC^0(\oplus)$ (the class of languages accepted by constant-depth, polynomial-size circuits of AND, OR, and PARITY gates)!

Q: Wow! Are you saying that it isn't even known if FACT is in $AC^0(\oplus)$ or not?

A: That's *not* what I said! Actually, it was shown in [BL87] that the number of 1's in a string $x = x_1 \ldots x_n$ is a multiple of 3 if and only if the number whose binary representation is $x_1 0 x_2 0 \ldots 0 x_n$ is a multiple of 3. Thus, if FACT were in $AC^0(\oplus)$, then certainly we could solve the Mod3 problem with $AC^0(\oplus)$ circuits, in contradiction to [Smo87]. In fact this shows that FACT is not in $AC^0(\mathrm{Mod}p)$ for any prime $p \neq 3$.

Q: So, is FACT in $AC^0(\mathrm{Mod}\ 3)$?

A: No. If you look at the finite automaton for the set of binary strings that are a multiple of 5, you will see that the number of 1's in a string $x = x_1 \ldots x_n$ is a multiple of 5 if and only if the number whose binary representation is $000x_1000x_2000\ldots000x_n$ is a multiple of 5. Thus, if FACT were in $AC^0(\mathrm{Mod}3)$, then certainly we could solve the Mod5 problem with $AC^0(\mathrm{Mod}3)$ circuits, again in contradiction to [Smo87]. Thus we have that FACT is not in $AC^0(\mathrm{Mod}p)$ for any prime p.

Q: So, we know that FACT is not *in* $AC^0(\mathrm{Mod}p)$ for any prime p, but we don't know if FACT is *hard* for any such class. How about $AC^0(\mathrm{Mod}6)$?

A: It remains an open question if there is anything in $\mathrm{NTIME}(2^n)$ that does not have linear-size, depth-three circuits of Mod6 gates.

Q: Thanks for reminding me just how little we know. What does this have to do with the question of whether FACT is complete for some complexity class?

A: I'm getting to that. If FACT were complete for some reasonable complexity class, then, by [AAR96], FACT would be isomorphic to $\mathrm{FACT}\times\{0,1\}^*$ (since $\mathrm{FACT}\times\{0,1\}^*$ would clearly also be complete for the same complexity class).

Q: This reminds me of something from my computability textbook [Ro67]. A set A that is recursively isomorphic to $A \times \{0,1\}^*$ is called a "cylinder".

A: Yes. This same terminology was imported to complexity theory in [MY85].

Q: Are we now ready for you to state your conjecture?

A: Yes.

> **Conjecture:** There is no isomorphism computable and invertible in depth-three AC^0 mapping $\mathrm{FACT}\times\{0,1\}^*$ onto FACT.

That is; FACT is not a (depth-three) AC^0 cylinder. If this conjecture is true, then factoring is not complete for any reasonable complexity class under AC^0 reductions.

Q: Isn't this conjecture just another example of an impossibly difficult question in complexity theory? Wouldn't it imply $\mathrm{NLOG} \neq \mathrm{P}$, or $TC^0 \neq \mathrm{NP}$, or something?

A: I don't think so. If I could show that A is not an AC^0 cylinder, where A is some set in NP that is hard for TC^0 (for instance), then I would know that TC^0 is contained in the class of languages that are reducible to A, which is properly contained in NP (since all NP-complete sets *are* AC^0 cylinders), and thus I could conclude that TC^0

is not equal to NP. However, since FACT is *not* known (or even believed) to be hard for TC^0, this obstacle is removed.

Q: What intuition do you have about why the conjecture might be true?

A: First, note that it is not too hard to show that FACT is a P-cylinder. That is, there *is* an isomorphism computable and invertible in polynomial time between FACT and FACT$\times\{0,1\}^*$. (Sketch of how to compute such an isomorphism: It is sufficient to give a length-increasing, invertible reduction from FACT$\times\{0,1\}^*$ to FACT; a result in [BH77] can be used to construct the isomorphism from this. Given (x,i,b) and y, we need to find some (x',i',b') such that $(x,i,b) \in F$ if and only if $(x',i',b') \in F$. We can directly compute the largest exponent e_j for each of the first $|y|$ primes, such that $p_j^{e_j}$ to that exponent divides x. Then, by doing some simple encoding using the exponents of the first $|y|$ primes, we can build a new number, whose prime factorization "encodes" the prime factorization of x, along with the bits of y. If we are careful about how we do this, then the bits of the prime factorization of x that deal with large primes are essentially unchanged (and thus i will be mapped to $i' = i + j$ for some easy-to-compute j). The details are straightforward but tedious.)

All of my attempts to build an isomorphism between FACT and FACT$\times\{0,1\}^*$ involve doing quite a bit of multiplication and division – which is impossible for an AC^0 circuit. One approach to try to prove the conjecture would be to show that *any* circuit computing such an isomorphism would be usable as an oracle to compute multiplication, or at least to compute parity infinitely often.

Q: Still, this seems like an ambitious project.

A: Perhaps. Certainly, showing that two sets are *not* isomorphic often seems to be quite difficult! (Note, however, that [AAIPR97] does contain results of this form. The sets SAT and SAT$'$ are not AC^0 isomorphic.)

Another approach would be to try to show that there is no length-increasing one-one function f in NC^0 reducing FACT$\times\{0,1\}^*$ to FACT. (Note that the "one-one" condition is crucial here. Otherwise, in NC^0 one could map (x,i,b,z) to $(x0^{|z|}, i', b)$ for appropriate (easy-to-compute) i'.) The results of [AAR96] can again be used to show that this implies that FACT is not complete.

Q: Do these observations apply to problems other than factorization?

A: They certainly do. I would guess that primality testing and computation of the greatest common divisor also fail to be complete for any reasonable complexity class, for the same reason.

It would also be worth trying to say something concrete about the multiplication problem. That is, consider the set MULT $= \{(x,y,i,b)$: the ith bit of the binary representation of $x \cdot y$ is $b\}$. MULT is certainly an important set, and in some sense its complexity is fairly well understood. It is in TC^0, and it is complete for TC^0 under AC^0-*Turing* reducibility. Since it is still an important open question if TC^0 is equal to NC^1, I believe it is still an open question whether MULT is complete for

NC^1 under AC^0 many-one reducibility. I think it is quite possible that one might be able to show that MULT is *not* complete for any reasonable complexity class (under AC^0 many-one reducibility).

Q: Wouldn't this allow one to prove that TC^0 is not equal to NC^1?

A: Not that I see. Note in this regard that it is already known that the MAJORITY function is not complete for NC^1 under AC^0 many-one reducibility. (That is because it is shown in [Be94] that there is no AC^0 many-one reduction from the PARITY language to MAJORITY.) MAJORITY and MULT have quite similar complexity, in that both are complete for TC^0 under AC^0-Turing reducibility.

Q: Is this the end of the article?

A: Almost. I just want to comment that we really do agree on one point.

Q: What is that?

A: Currently, complexity theory has no credible approach toward trying to show that factorization is difficult to compute. The only real evidence of intractability that complexity theory offers currently is based on reducibility and completeness. Factorization seems to fall outside that framework. With luck, however, it might be possible to *prove* that this is the case.

References

[AA96] M. Agrawal and E. Allender, *An Isomorphism Theorem for Circuit Complexity*, in Proc. 11th Annual IEEE Conference on Computational Complexity (1996) pp. 2–11.

[AAIPR97] M. Agrawal, E. Allender, R. Impagliazzo, T. Pitassi, and S. Rudich, *Reducing the complexity of reductions*, in Proc. 29th ACM STOC (1997), pp. 730–738.

[AAR96] M. Agrawal, E. Allender and S. Rudich, *Reductions in Circuit Complexity: An Isomorphism Theorem and a Gap Theorem*, to appear in J. Comp. Sys. Sci. Some of the results in this paper were originally announced in [AA96] and [AAIPR97].

[ABI93] E. Allender, N. Immerman, and J. Balcázar, *A first-order isomorphism theorem*, SIAM Journal on Computing **26** (1997) 557–567.

[Al97] E. Allender, *Some pointed questions concerning asymptotic lower bounds*, Computational Complexity Column, Bulletin of the EATCS 62 (1997).

[AR98] E. Allender and K. Reinhardt, *Isolation, Matching, and Counting*, in Proc. 13th Annual IEEE Conference on Computational Complexity, 1998, pp. 92–100

[Ba89] D. Barrington, *Bounded-width polynomial size branching programs recognize exactly those languages in NC^1*, Journal of Computer and System Sciences 38 (1989), 150–164.

[Be94] R. Beigel, *When do extra majority gates help? Polylog(n) majority gates are equivalent to one*, Computational Complexity **4** (1984) 297–388.

[BH77] L. Berman and J. Hartmanis, *On isomorphism and density of NP and other complete sets*, SIAM J. Comput. **6** (1977) 305–322.

[BL87] Ravi Boppana and Jeff Lagarias, *One-Way Functions and Circuit Complexity*, Information and Computation **74** (1987) 226-240.

[Bu87] S. Buss, *The Boolean Formula Value Problem is in ALOGTIME*, in Proc. 19th ACM Symposium on Theory of Computing, pp. 123-131, 1987.

[Fe92] M. Fellows and N. Koblitz. *Self-witnessing polynomial-time complexity and prime factorization* in *Proc. of the 7th IEEE Structure in Complexity Conference*, pages 107–110, 1992.

[FFK96] S. Fenner, L. Fortnow, and S. Kurtz, *The isomorphism conjecture holds relative to an oracle*, SIAM Journal on Computing **25** (1996) 193-206.

[GHR95] R. Greenlaw, J. Hoover, and W.L. Ruzzo. *Limits to Parallel Computation: P-Completeness Theory.* Oxford University Press, 1995.

[GJ79] Michael R. Garey and David S. Johnson, *Computers and Intractability,* Freeman, 1979.

[Ha78] Juris Hartmanis, *On the logtape isomorphism of complete sets,* Theoretical Computer Science **7** (1978) 273-286.

[JPY94] D. Joseph, R. Pruim, and P. Young, *Collapsing degrees in subexponential time*, Proc. 9th Structure in Complexity Theory Conference (1994) pp. 367–382.

[JY85] D. Joseph and P. Young, *Some remarks on witness functions for non-polynomial and non-complete sets in NP*, Theoretical Computer Science **39** (1985) 225–237.

[KLD86] Ker-I Ko, Timothy J. Long, and Ding-Zhu Du, *On one-way functions and polynomial-time isomorphisms*, Theoretical Computer Science **47** (1986) 263–276.

[KMR90] S. Kurtz, S. Mahaney, and J. Royer, *The structure of complete degrees*, in A. Selman, editor, *Complexity Theory Retrospective*, Springer-Verlag, 1990, pp. 108–146.

[KMR95] S. Kurtz, S. Mahaney, and J. Royer, *The isomorphism conjecture fails relative to a random oracle*, J. ACM **42** (1995), 401–420.

[KST93] J. Köbler, U. Schöning, and J. Torán, *The Graph Isomorphism Problem; It's Structural Complexity*, Birkhäuser, 1993.

[KUW86] R. Karp and E. Upfal and A. Wigderson, *Constructing a perfect matching is in random NC*, Combinatorica **6** (1986) 35–48.

[La75] R. Ladner. *On the structure of polynomial-time reducibility*, J. Assn. Comp. Mach. **22** (1975) 155–171.

[MY85] S. Mahaney and P. Young, *Reductions among polynomial isomorphism types*, Theoretical Computer Science **39** (1985) 207-224.

[Ni92] Noam Nisan, *Using Hard Problems to Create Pseudorandom Generators*, MIT Press (1992).

[Ro67] H. Rogers, *Theory of Recursive Functions and Effective Computability*, McGraw-Hill, 1967.

[Ro95] J. Rogers, *The isomorphism conjecture holds and one-way functions exist relative to an oracle*, J. Computer Sys. Sci. **54** (1997) 412–423.

[Smo87] R. Smolensky. *Algebraic methods in the theory of lower bounds for Boolean circuit complexity*, in *Proceedings, 19th ACM Symposium on Theory of Computing*, pages 77–82, 1987.

[To98] J. Torán, talk presented at the 1998 IEEE Conference on Computational Complexity, Buffalo, NY.

[Yo90] Paul Young, *Juris Hartmanis: Fundamental Contributions to Isomorphism Problems,* in *Complexity Theory Retrospective,* Alan Selman, ed., Springer-Verlag (1990), 28-58.