Research Statement

Kerry Ojakian

I work in logic and computation. My research in logic has involved formalizing finite combinatorics in the logical theory of Bounded Arithmetic (a weak subtheory of Peano Arithmetic). Within computation I have focused on computability over the real numbers, in particular on Computable Analysis and its relationship to analog models of computation. In addition, I have a strong interest in combinatorics, and plan to work in this area. I now discuss these 3 areas (preprints of my papers are available at my web site: http://www.math.ist.utl.pt/∼ojakian/).

Real Computation

My work in real computation (joint with Manuel Campagnolo) has involved showing that functions of *Computable Analysis* can be characterized by alternative *analog-time* models of computation. Numerous models of computability for functions on the real numbers have been developed, each one making a claim to which set of functions over the real numbers are *computable*. In Computable Analysis a standard discrete Turing Machine is used to define a notion of computation: A function over the reals is computable (in the sense of Computable Analysis) if from rational approximations to the real number inputs, a rational approximation to the output can be computed using a Turing Machine (see Grzegorczyk [10], Ko [11] and Weihrauch [19]). In Computable Analysis standard Turing Machines are used and so time proceeds in a discrete step by step manner. In the analog models, time proceeds in a continuous (or analog) manner. The analog model we have focused on is Moore's *Real Recursive Functions* [13]. These classes of functions are defined analogously to the *recursive functions* over the naturals: The recursive functions are a function algebra defined by presenting some simple basic functions, and closing the functions under some operations, one of which is *primitive recursion*. In the case of the Real Recursive Functions, we begin with basic functions over the reals and replace the primitive recursion operation by an operation which allows us to solve a differential equation. There are a number of goals motivating our research. On the one hand, these kinds of results are interesting because they show that two different approaches to computation (discrete-time versus analog-time) can be made equivalent. More broadly, these results can be seen as part of a search for a *Church-Turing thesis* in the context of real computation. For computation on the natural numbers there is an agreed upon notion of what it means to be *computable*, in a large part due to the impressive fact that a number of different models of computation yield the same set of functions. Developing similar correspondences for real computation should be significant. A final motivation for this work is the possibility of shedding light on the questions of classic discrete computational complexity theory. Costa and Mycka [9] have work that provides an *analytic condition* (e.g. a statement about differential equations) that is equivalent to $\mathsf{P} = \mathsf{NP}$. One of our hopes is to obtain useful analytic conditions equivalent to classic complexity questions; then techniques from analysis could conceivably be used to answer questions in discrete

complexity theory.

There has been significant recent work relating analog models to Computable Analysis. Bournez and Hainry [1] characterized the $\mathcal{C}^2$ (twice continuously differentiable) *elementary computable* functions of Computable Analysis with a function algebra (i.e. using the model of Real Recursive Functions). This work was extended to the *computable* functions of Computable Analysis in Bournez and Hainry [2]. We have extended the work on the elementary computable functions, obtaining alternative proofs, removing the restriction to $\mathcal{C}^2$, and showing that fewer basic functions suffice (see our papers [7] and [5]). We have also proved a variant of their result for the computable functions of Computable Analysis, providing what we believe is a simpler function algebra. We replace two of their operations by the single natural operation allowing us to solve an ordinary differential equation (see our paper [6]).

Significantly, our work introduces new general techniques which we call *approximation and lifting* (see our papers [4] and [7]). Suppose $\mathcal{CA}$ is the set of functions of Computable Analysis and $\mathcal{FA}$ is some set of functions given by a function algebra; our goal is to show $\mathcal{CA} = \mathcal{FA}$. In the previous work of others, a central part of the proof involved showing that a Turing Machine computing a function in $\mathcal{CA}$ can be simulated in the class $\mathcal{FA}$. We avoid such a simulation, by starting with a classic result on the naturals that says something analogous to $\mathcal{CA} = \mathcal{FA}$. Then through a somewhat involved *lifting process*, we show how the result on the naturals can be lifted up to the rationals and eventually up to the reals. A fundamental aspect of this approach is the notion of an *approximate equality*, a property that can hold between classes of functions. Supposing $\mathcal{A}$ and $\mathcal{B}$ are classes of functions, we define an approximate equality, writing it as $\mathcal{A} \approx \mathcal{B}$; it means (roughly) that any function from one class can be *approximated* by some function from the other class. Our proof roughly proceeds as follows: First we show a sequence of approximate equalities $\mathcal{CA} \approx \mathcal{H}_1 \approx \ldots \approx \mathcal{H}_k \approx \mathcal{FA}$, for some well-chosen intermediary classes of functions $\mathcal{H}_i$, then by *transitivity* of the approximate equality we conclude $\mathcal{CA} \approx \mathcal{FA}$, and finally, because the approximation is sufficiently strong, we obtain our goal of $\mathcal{CA} = \mathcal{FA}$.

The advantages of our approach are that on the one hand it provides a method of proof which avoids a Turing Machine simulation, while on the other hand it should be much more amenable to generalization and broader application. A number of the lemmas used in our proof about the elementary computable functions are general (i.e. not pertaining specifically to elementary computability), and others should be generalizable. The approximation notions provide a unifying language for discussing various results in the area (see our paper [5]). Under the right conditions, the approximate equality is transitive, thus allowing us to break down one approximate equality into a series of smaller and more natural tasks (as was illustrated above in showing $\mathcal{CA} \approx \mathcal{FA}$).

Real Computation: Future Work and Work in Progress

We have work in progress that improves our results in [6], which characterized the $\mathcal{C}^2$ functions of Computable Analysis. We believe we can remove the restriction to the $\mathcal{C}^2$ functions. We also hope to simplify the set of basic functions.

A big goal for us and others working in this area is to pursue the characterization of weaker complexity classes of Computable Analysis, since the work to date has only pertained to elementary computability or stronger. We have work in progress on the *counting hierarchy*. It would also be interesting to consider classes such as the the polynomial-space functions, and of course, eventually the polynomial-time functions.

In all our work, we intend to use our techniques of *approximation and lifting*, but we would like to push this even further by working out a general theory. In our work to date, parts have been general and other parts of it have been specific to the various applications. Though we claim it is amenable to generalization, a general theory has not yet been worked out. We have begun to work in this direction, expecting it to facilitate future work of this sort, at least being a very useful tool, and possibly providing a deep connection between computability on the naturals and computability on the reals.

Bounded Arithmetic

My work in *Bounded Arithmetic* (concerning Bounded Arithmetic see Buss [3] and Krajíček [12]) has involved formalizing various *probabilistic methods* and *linear algebra methods* (with a focus on applications to *Ramsey theory*); by *formalizing* I mean to carry out a proof formally in some logical theory (my work appears in my paper [15] and my thesis [14]). My work is a kind of *Reverse Mathematics* for finite combinatorics. The goal of Reverse Mathematics is to categorize mathematical theorems according to the axioms needed to prove them (see Simpson [17]). Theories of Bounded Arithmetic are of just the right strength for the formalization of much of finite combinatorics due to the connection between Bounded Arithmetic and the polynomial-time hierarchy. Typically, what combinatorialists call *constructive proofs* exhibit the object explicitly (for example, explicitly giving the coloring of the edges of a graph) and can be formalized with *polynomial-time reasoning*, i.e. in a particular weak subtheory of Bounded Arithmetic. What are referred to as *non-constructive proofs*, such as probabilistic arguments, often naturally formalize in a stronger subtheory of Bounded Arithmetic, i.e. corresponding to reasoning higher up in the polynomial-time hierarchy. Thus, Bounded Arithmetic provides a hierarchy of theories of the correct strength to make formal the distinctions sometimes made informally by combinatorialists.

Combinatorics using probabilistic and linear algebra methods typically cannot formalize into Bounded Arithmetic in a direct manner because the objects used in the proof are "too large." My formalizations have involved coming up with alternative proofs. Often we can then isolate some particular axiom, say A, as being of particular importance in proving formally some theorem T, or in other words, we can say that axiom A implies theorem T. In some cases I have shown that A is necessary to prove T by showing a *reversal*, namely, that the theorem T implies the axiom A.

A linear algebraic proof typically works by associating the combinatorial objects in question with vectors and then extracting information from the fact that the vectors are linearly independent. A question asked by combinatorialists is whether certain claims proved using linear algebra can be proved without it. I have begun to answer this kind of question, introducing a linear algebra axiom related to linear independence (related to the work of Soltys and Cook [18] on theories of feasible linear algebra). I show that some results formalize in a weak theory of Bounded Arithmetic, along with this axiom, while other results can be formalized with alternative proofs that avoid this axiom. In these latter cases, I show that instead, a simpler axiom called the *weak pigeonhole principle* (WPHP) suffices (it essentially states that there does not exist an injective function from a set of size $2n$ to a set of size n). Furthermore I show that the WPHP is necessary, since I can obtain reversals, namely that not only are the combinatorial facts implied by the WPHP, but the facts also implies the WPHP.

A probabilistic method proof works by considering an appropriate probability space and extracting combinatorial information (usually the existence of some object, like a graph coloring)

from a probabilistic property. A probabilistic method proof, such as lower bounds on Ramsey numbers, cannot be formalized directly in Bounded Arithmetic because it refers to exponentially large sets like the sample space. My work in formalizing various arguments of this kind has avoided the direct argument, instead using the WPHP to simulate the argument. For the case of lower bounds on Ramsey numbers I have not obtained reversals (and I expect this would be difficult). However, in the case of upper bounds on Ramsey numbers, Pudlák [16] essentially showed that the WPHP implies a Ramsey upper bound, while I obtained a reversal, showing that the Ramsey upper bound implies the WPHP. I observed a similar equivalence between a stronger Ramsey upper bound statement and the axiom stating that the exponential function is defined everywhere.

Bounded Arithmetic: Future Work and Work in Progress

I intend to develop more fully the programme of Reverse Mathematics for finite combinatorics in Bounded Arithmetic, formalizing proofs and proving reversals. To facilitate this programme, I have work in progress on a *type theory* for Bounded Arithmetic (with a different motivation than Cook and Urquhart's feasible type theory [8]; e.g. while my functions are intended to be finite, and thus more typical for Bounded Arithmetic, their functions are intended to be infinite). When working in a first-order Bounded Arithmetic, the objects of the theory are natural numbers, thus forcing us to code all our mathematical objects as numbers. Once completed, my type theory should allow us to avoid these cumbersome codings of combinatorial objects (such as an edge coloring of a graph), letting us work more directly with the objects in question. This should be particularly useful in very weak subtheories of Bounded Arithmetic where the coding becomes non-trivial. Another motivation for my type theory is an attempt to unify the various theories of Bounded Arithmetic (i.e. there are first, second, and third order theories), so that rather than proving similar results in each system (such as the polynomial witnessing claim), the result could be proven once and for all in the type theory, with the analogous results in the other theories following as relatively simple corollaries.

Combinatorics

Though I do not have research directly in combinatorics, my work in logic (as discussed in the previous section) touches on Ramsey theory, probabilistic methods, and linear algebra methods. I am very interested in complementing this work in logic with work which lies more directly within combinatorics. During my short post-doc in Prague, I worked on constructive Ramsey lower bounds, though I stopped this research when I moved to a new post-doc position in Lisbon. I am now very serious about getting back into combinatorics. While I generally like all areas of combinatorics, I have begun reading in Ramsey theory and in design theory.

References

[1] O. Bournez and E. Hainry. Elementarily computable functions over the real numbers and $\mathbb{R}$-sub-recursive functions. *Theoretical Computer Science*, 348(2–3):130–147, 2005.

[2] O. Bournez and E. Hainry. Recursive analysis characterized as a class of real recursive functions. *Fundamenta Informaticae*, 74(4):409–433, 2006.

[3] S. Buss. *Bounded Arithmetic*. Bibliopolis, Italy, 1986.

[4] M. L. Campagnolo and K. Ojakian. The methods of approximation and lifting in real computation. In Douglas Cenzer, Ruth Dillhage, Tanja Grubba, and Klaus Weihrauch, editors, *Proceedings of the Third International Conference on Computability and Complexity in Analysis, CCA 2006, Gainesville, Florida, USA, November 1–5, 2006*, volume 167 of *Electronic Notes in Theoretical Computer Science*, Amsterdam, 2007. Elsevier.

[5] M. L. Campagnolo and K. Ojakian. Using approximation to relate computational classes over the reals. In J. Durand-Lose and M. Margenstern, editors, *MCU 2007*, volume 4664 of *Lecture Notes in Computer Science*, pages 39–61. Springer-Verlag, 2007.

[6] M. L. Campagnolo and K. Ojakian. Characterizing computable analysis with differential equations. In *Fifth International Conference on Computability and Complexity in Analysis*, Electronic Notes in Theoretical Computer Science, 2008. to appear.

[7] M. L. Campagnolo and K. Ojakian. The elementary computable functions over the real numbers: applying two new techniques. *Archives for Mathematical Logic*, 46:593–627, 2008.

[8] S. Cook and A. Urquhart. Functional interpretations of feasibly constructive arithmetic. *Annals of Pure and Applied Logic*, 63:103–200, 1993.

[9] J. F. Costa and J. Mycka. The $P \neq NP$ conjecture in the context of real and complex analysis. *Journal of Complexity*, 22(2):287–303, April 2006.

[10] A. Grzegorczyk. Computable functionals. *Fund. Math.*, 42:168–202, 1955.

[11] K.-I. Ko. *Complexity Theory of Real Functions*. Birkhaüser, 1991.

[12] J. Krajíček. *Bounded Arithmetic, Propositional Logic, and Complexity Theory*. Cambridge University Press, New York, 1995.

[13] C. Moore. Recursion theory on the reals and continuous-time computation. *Theoretical Computer Science*, 162:23–44, 1996.

[14] K. Ojakian. *Combinatorics in Bounded Arithmetic*. PhD thesis, Carnegie Mellon University, 2004. Supervised by J. Avigad.

[15] K. Ojakian. Upper and lower Ramsey bounds in bounded arithmetic. *Annals of Pure and Applied Logic*, 135(1-3):135–150, September 2005.

[16] P. Pudlák. Ramsey's theorem in bounded arithmetic. In E. Borger, editor, *Lecture Notes in Computer Science*, volume 533, pages 308–312. Springer-Verlag, 1991.

[17] S. Simpson. *Subsystems of Second Order Arithmetic*. Springer-Verlag, New York, 1999.

[18] M. Soltys and S. Cook. The proof complexity of linear algebra. *Annals of Pure and Applied Logic*, 130:277–323, 2004.

[19] K. Weihrauch. *Computable Analysis: An Introduction*. Springer-Verlag, 2000.