2008 CRM-Fields-PIMS Prize: Allan Borodin

Stephen Cook

Allan Borodin of the University of Toronto has been awarded the 2008 CRM-Fields-PIMS prize. According to the citation "*Professor Borodin is a world leader in the mathematical foundations of computer science. His influence on theoretical computer science has been enormous, and its scope very broad. Jon Kleinberg, winner of the 2006 Nevanlinna Prize, writes of Borodin, "he is one of the few researchers for whom one can cite examples of impact on nearly every area of theory, and his work is characterized by a profound taste in choice of problems, and deep connections with broader issues in computer science." Allan Borodin has made fundamental contributions to many areas, including algebraic computations, resource tradeoffs, routing in interconnection networks, parallel algorithms, online algorithms, and adversarial queuing theory.*"

Borodin received his B.A. in Mathematics from Rutgers University in 1963, his M.S. in Electrical Engineering & Computer Science in 1966 from Stevens Institute of Technology, and his Ph.D. in Computer Science from Cornell University in 1969. He was a systems programmer at Bell Laboratories in New Jersey from 1963-1966, and a Research Fellow at Cornell from 1966-1969. Since 1969 he has taught with the computer science department at the University of Toronto, becoming a full professor in 1977, and chair of the department from 1980-1985. He has been the editor of many journals including the SIAM Journal of Computing, Algorithmica, the Journal of Computer Algebra, the Journal of Computational Complexity, and the Journal of Applicable Algebra in Engineering, Communication and Computing. He has held positions on, or been active in, dozens of committees and organizations, both inside and outside the University, and has held several visiting professorships internationally. In 1991 Borodin was elected a Fellow of the Royal Society of Canada.

The discipline of computer science has been an exceptionally successful blend of engineering and mathematical science with a healthy dose of human factor and aesthetic issues. Allan Borodin has made significant contributions to many diverse aspects of the discipline, with a major focus on the more mathematical areas. A common theme in his research is that he explores fundamental questions that should have well-understood explanations but seem to often defy answers to even the most basic forms of these questions. As a result Borodin has often been at the forefront of developing new models and problem formulations that have become standard frameworks for studies in computer science.

Perhaps the most basic scientific aspect of computer science is to understand the intrinsic limitations of what can and what cannot be *efficiently* computed in various models of computing with respect to various measures of complexity. This study is the heart of complexity theory. The other side of the complexity theory coin is the design and analysis of algorithms. Borodin has been involved in both sides of this coin since his first publication in 1969. In his Cornell PHD thesis, Borodin studied the time complexity classes introduced by Hartmanis and Stearns and the more abstract complexity measures axiomatized by Blum. He showed that "constructible" bounding functions as used by Hartmanis and Stearns to develop complexity hierarchies are necessary in the sense that for any complexity measure (be it time, space, etc.) there are arbitrarily large gaps (where no new functions are being computed) created by non-constructible bounding functions [Bor72]. Another thesis result (with Constable and Hopcroft) showed that time complexity classes are dense [BCH69].

Borodin soon became more focused on the complexity of specific functions and, in particular, what we now call "algebraic complexity theory". The complexity world was basically unchartered territory at the end of the 1960s although many surprising and widely applicable results (for example, the Fast Fourier Transform and fast integer multiplication) were developed outside the the confines of a formal theory. A number of results accelerated the development of complexity theory. One such result was Cook's formulation of the class NP and the identification of NP complete problems which became and still remains the main source of evidence that many common combinatorial problems cannot be solved efficiently (i.e. within polynomial time). On the other side of the coin, Strassen's surprising result that matrix multiplication can be computed within $O(n^{\log_2 7}) \approx O(n^{2.81})$ arithmetic operations showed that our common intuition and beliefs cannot be trusted (the obvious method requires n^3 multiplications). Following Strassen's dramatic result, Borodin proved a number of results helping to establish the field of algebraic complexity. Resurrecting an old question raised by Ostrovsky, Borodin showed that Horner's rule for evaluating a polynomial is uniquely optimal in being the only method that can achieve the optimal $2n$ arithmetic operations. Since (even with preconditioning) $n/2$ multiplications/divisions and n additions/subtractions are required for one polynomial evaluation for most degree n polynomials, how many operations are needed to evaluate a degree n polynomial at (say) n arbitrary points? When the evaluation points are the powers of a suitable primitive root of unity, the FFT performs these evaluations in $O(n \log n)$ operations rather than the $O(n^2)$ operations required if one evaluates at each point individually. By reduction to Strassen's fast matrix multiplication, Borodin and Munro showed that $O(n^{1.91})$ operations are sufficient

[BM71]. Then Borodin and Moenck showed that $\Omega(n \log n)$ non scalar multiplications and $O(n \log^2 n)$ total operations are sufficient [BM74], which remains the best asymptotic bound for total operations (and matched by Strassen's algebraic geometry based $\Omega(n \log n)$ lower bound for the number of non scalar multiplications). The Strassen bound uses the Bezout Theorem on the degree of an algebraic variety to generalize the obvious fact that an n^{th} degree polynomial requires $\log n$ multiplications (since the degree can at most double following a multiplication).

Using the FFT, two n^{th} degree polynomials can be multiplied in $O(n)$ non scalar multiplications and $O(n \log n)$ additions. Is there an analogue to the degree bound so as to establish lower bounds on the number of additions to compute polynomials? Borodin and Cook [BC76] show that the number of *real* roots of a polynomial is bounded by the minimal number of additions used to compute the polynomial. The Borodin and Cook lower bounds were improved by Risler using results from real algebraic geometry. Beyond these research contributions, Borodin and his first PHD student Ian Munro wrote the seminal text book [BM75] in the area of algebraic complexity and it remained the most authoritative source for approximately 20 years.

Another area of interest for Borodin concerns parallel computation and network routing. How does parallel time complexity relate to the more standard complexity measures of time and space? Following the known results relating sequential time with uniform circuit size, Borodin showed that the space measure is directly related to uniform circuit depth, a basic measure of parallel complexity. Unlike the situation for classical sequential time studies, there are alternative models of parallel computation, including various parallel RAM models and interconnection network models. In order for an interconnection network to be able to simulate a RAM, the network must be able to simply and efficiently rout simultaneous messages. Oblivious routing schemes are simple in the sense that the path of each message is independent of the routes of other messages. Valiant showed that by obliviously routing to a random intermediate node, any permutation could be routed in time $O(d)$ on a d-dimensional hypercube. This is asymptotically optimal since d is the diameter of the network. Borodin and Hopcroft [BH85] showed that this use of a random intermediate node is necessary in the following strong sense: in any degree d network with n nodes, for any deterministic (i.e. non randomized) oblivious routing algorithm, there exists a permutation that will have a bottleneck node forcing the routing to take at least $\Omega\sqrt{n}/d^{3/2}$ time. Borodin was also the co-designer of some surprising parallel algorithms, including (with von zur Gathen and Hopcroft) [BvzGH82] a randomized parallel greedy algorithm to derive a $\log^2 n$ parallel time (i.e. depth of arithmetic circuit) algorithm for computing the rank of an $n \times n$ matrix, and a $\log \log n$ algorithm for merging two lists on a CREW (Concurrent Read Exclusive Write) RAM model.

The work on packet routing led to a new area of research. Packet routing can be viewed as a queuing system in which the edges of the network become the processes and one can study the queueing effects in terms the nature of the network and/or the scheduling rules used by the nodes of the network. In this setting, input requests (e.g. oblivious packet paths or requests for packet transmission along any path from source to target) are characterized more by burstiness rather than by any standard probabilistic distribution. Furthermore, the processing time (transmission of a single or a bounded number of packets along an edge) is usually considered to have a well-defined time. Borodin, Kleinberg, Raghavan, Sudan and Williamson [BKR$^+$01] modeled this burstiness by an adversarial model and developed an area named "adversarial queuing theory". There are some natural queuing limitations on the stability (i.e. bounded queue sizes and time to complete a transmission) with the main limitation (say in oblivious routing) being that the rate of requests for an edge cannot exceed the processing rate of the edge. Borodin et al began the study as to which networks are always stable at a given rate (independent of the scheduling rule) and which scheduling rules are always stable (independent of the network). Adversarial studies of packet routing and other queueing systems has led to a number of surprising results (e.g. the instability of FIFO at any rate for certain networks as shown by Bhattacharjee and Goel).

While complexity theory has been very successful in many aspects (e.g. understanding the relation between complexity measures, establishing complexity based cryptography, utilizing hardness to develop pseudo random generators, the development of new notions of "proofs" including interactive proofs and probabilistically checkable proofs), the major limitation of the field thus far is in the inability to prove complexity impossibility results for "explicitly defined natural problems" (for example, NP search and optimization problems). More specifically, non-linear time bounds (on a sufficiently general model of computation) or space bounds greater than $\log n$ still elude us. Perhaps then the simplest barrier to break is to exhibit a problem which cannot be simultaneously computed in small time and space. Borodin lead a group of coauthors [BFK$^+$81] to prove the first time-space tradeoff result for comparison based sorting in what can be said to be the most general model for such a result. They considered comparison branching programs which are DAGS where nodes are labelled by comparisons "$a_i \leq a_j$?" between elements from a given "read-only" input set of n elements. In this model, edges are labelled by sequences of input elements that are being output if this edge is traversed. The complete sequence of outputs along any path defines the output of the program. In this non-uniform model (like circuits, a different program is allowed for each n), time is the length of the longest path (or expected path length if one were considering average case complexity), and space is the logarithm of the number of nodes in the DAG (i.e. the information theoretic lower bound on the memory being used). In contrast to list merging, which can be computed simultaneously in linear time and $O(\log n)$ space, Borodin et al show that the time space product $T \cdot S = \Omega(n^2)$; that is, any small space method must require significantly more time than the optimal $n \log n$ bound achievable by methods such as merge-sort. (For all space bounds $S(n)$ between $\log n$ and n, a corresponding upper bound can be obtained.)

This paper [BFK$^+$81] was seminal and started a long and continuing research effort to derive time space bounds for natural problems in appropriate models. The sorting tradeoff was soon followed by a similar comparison branching program tradeoff for a decision problem, namely the element distinctness problem. The initial element distinctness tradeoff was by Borodin et al [BFadH$^+$87], and it was then improved by Yao. These comparison branching programs (where the algorithm does not have access to the encoding of the input elements) leaves open the possibility that the corresponding Boolean problems (e.g. say encoding integer inputs in binary) can be computed using simultaneously small time and space. This consideration led Borodin and Cook [BC82] to introduce the R-way branching program model, where now inputs are considered to be inputs in some range $[1, R]$, and branching program nodes are of the form "$a_i =?$", with up to R branches corresponding to each of the possible values of a_i. Time and space are defined as before. Borodin and Cook showed that sorting n numbers in the range $[1, n^2]$ required $T \cdot S = \Omega(n^2)$, proving a very strong tradeoff result, since the total binary encoding length of the input is only $O(n \log n)$ bits. This represents the first negative result for an explicit (polynomial time computable) Boolean problem in a completely general model, albeit not a decision problem. It took approximately 20 more years to establish negative results (of a much weaker form) for a Boolean decision problem.

In the mid 1980s, Borodin began working on the topic of online approximation algorithms which became known as competitive analysis, whereby the performance of an online algorithm (making decisions for each input as it arrives) is compared to the performance of an optimal solution with complete knowledge of the entire input. There had been a number of earlier important results concerning online algorithms for specific problems that need not necessarily be considered as online problems (for example, Graham's study of the makespan problem, Kierstead and Trotters online interval colouring, and Yao's study of online bin packing). Sleator and Tarjan proposed competitive analysis (in contrast to distributional studies) for problems which were inherently online such as paging and list accessing. Borodin, Linial and Saks [BLS92] then proposed an abstract online problem framework called metrical task systems (MTS) which was soon followed by the k-server model of Manasse, McGeouch and Sleator. The introduction of competitive analysis for online problems and these abstract problem formulations spawned a wealth of research activity that has had an impact well beyond online problems. For example [BLS92] provides an optimal $2n - 1$ competitive ratio bound for deterministic algorithms for any n-state MTS. It also introduced randomized algorithms in this context showing that the uniform metric system had a $2H_n \approx 2 \ln n$ randomized competitive ratio. This led the way to a randomized paging algorithm by Fiat et al and, moreover, led to interest in trying to derive randomized algorithms for general MTS and k-server problems. In this context Bartal introduced Hierarchically Separated Tree spaces (HSTs) for which $O(\log n)$ randomized algorithms exist and furthermore arbitrary metric spaces can be efficiently embedded into such HSTs. The use of HSTs has now become a standard tool in combinatorial approximation.

Beyond the seminal MST work, Borodin was influential in a number of central results concerning online competitive analysis. Borodin et al [BIRS95] introduced a variant of competitive analysis so as to model the locality of reference exhibited by (for example) paging requests. Another landmark paper introduces "request-answer games", which provide a framework for defining most known online problems. In this very abstract setting (which, for example, includes the MTS and k-server settings), Borodin and coauthors [BDBK$^+$94] relate the power of different adversarial models for randomized online algorithms; namely, they identify the more standard oblivious adversary (as used in offline computation) where the adversary generates the input request sequence without knowledge of the algorithm's coin tosses, and adaptive adversaries where the adversary adaptively creates the input sequence by observing the coin tosses and actions of the online algorithm. For adaptive adversaries, the adversary (acting also as the "optimal benchmark") can either play the game online or play the game in hindsight as an offline player. A number of randomized algorithms were being studied relative to (not so precisely defined) adaptive adversaries. Ben David et al show that algorithms competing against online adaptive adversaries can be simulated by algorithms competing against offline adaptive adversaries which in turn can be simulated by deterministic algorithms thereby showing that randomization can only yield significantly improved competitive ratios when formulated as algorithms competing against oblivious adversaries.

Finally one of Borodin's most influential contributions to online analysis is his text [BEY98] with former student Ran El-Yaniv. The text (published in 1998) remains the authoritative reference for this area, although many significant results have followed its publication, including a number of results addressing questions raised in the book.

Borodin has made significant contributions to a number of other aspects of algorithm analysis. One paper with Ostrovsky and Rabani [BOR03] provides the first memory-search time results for problems (e.g. nearest neighbour and partial match search) in high dimensional spaces, proving that for deterministic algorithms some form of exponential "curse of dimensionality" must exist for a widely studied geometric search model.

Borodin's most recent research area has been an area he has essentially been creating, namely the attempt to study the power and limitations of "simple algorithms", especially (to date) for search and optimization problems. While we equate efficient algorithms with time and/or memory efficiency, there are other important aspects to algorithm design. It is a rather remarkable fact that for over 70 years we have a well-accepted formalization (i.e. the Church -Turing thesis) for the intuitive concept of "computable function" and the associated concept of an algorithm. And if we stay within classical computing models (in contrast to say quantum computers) we have a reasonably well-accepted definition of "efficiently computable". But we often want simple understandable algorithms, at least as starting points or benchmarks for developing more sophisticated, complex algorithms. That is, we tend to use a small set of basic algorithmic paradigms as a "toolbox" for an initial (and sometimes the best known or even optimal) method for solving large classes of problems in many settings. These basic paradigms include greedy algorithms, divide and conquer, dynamic programming, local search, primal dual algorithms and IP/LP rounding. Surprisingly, although we intuitively understand what these concepts mean, rarely do we attempt any precise formulation, and a precise formulation is necessary if one is to gain any insight into the ultimate power and limitations of these methods.

The long standing and significant study and use of greedy algorithms provides a great example of an algorithmic paradigm that seems so natural and obvious that no definition seems necessary. It is hard to think of a computational area where some concept of greediness does not appear. The elegant results of Edmonds, Korte, Lovasz connecting matroids and greedoids with the optimality of "the" natural greedy algorithm for certain set systems was the starting point for a number of insightful results concerning optimal greedy algorithms. But greedy algorithms are mainly used as a heuristic or to obtain approximation results. Borodin, Nielson and Rackoff [BNR03] introduce the priority algorithm framework as a model for "greedy-like" optimization algorithms in almost any setting. We can think of this framework as an offline extension of online algorithms. An input to a problem is a set of items (for example, jobs in a scheduling problem, vertices in a graph problem, propositional variables in a SAT problem) and a priority algorithm considers and makes decisions about items one by one but now in an order determined (in advance or adaptively) by the algorithm rather than the order given by (adversarial) nature. Of course, the trick here is formulate what orderings a "reasonable" algorithm can use. For example, it would make no sense to allow the algorithm to compute an optimal solution and a corresponding optimal order that allows the algorithm to produce the optimal solution. One approach would be to resort to complexity considerations and say that each item is chosen within some acceptable time. But that would bring us back to our current inability to prove limitations based on time complexity. Instead the priority framework relies on a simple to state concept of a local ordering. In fact, the allowable orderings are (at each iteration in the case of adaptive priority algorithms) those satisfying Arrow's IIA (independence of irrelevant attributes) axiom. Whereas in social choice theory this axiom is controversial, for greedy-like algorithms the concept allows great generality while still being amenable to analysis. And what does this have to do with greediness? In the priority framework it is not the ordering decisions that are greedy but rather (for greedy priority) it is the decisions being made for each input item that can be construed as greedy (say in the sense of "living for today") with respect to the given objective function. There are a number of results showing the limitations of such priority algorithms in different domains, starting with the initial scheduling results of Borodin, Nielson and Rackoff.

The priority framework is also the starting point for more powerful paradigms, such as some simple forms of primal dual algorithms using a reverse delete step, simple dynamic programming and backtracking. For example, the work of Borodin and coauthors [ABBO$^+$05] shows why DPLL style backtracking algorithms cannot solve 3SAT search and has limits to approximating Max2Sat but can solve 2SAT. They also show that the form of dynamic programming used for interval scheduling and knapsack algorithms have limitations. In particular, optimal dynamic programming algorithms for weighted interval scheduling on m machines must suffer a curse of dimensionality with respect m.

This recent algorithmic design work reflects the style of an extraordinarily productive and creative career.

References

[ABBO+05] M. Alekhnovich, A. Borodin, J. Buresh-Oppenheim, R. Impagliazzo, A. Magen, and T. Pitassi. Toward a model for backtracking and dynamic programming. In *Proceedings of Computational Complexity Conference (CCC)*, pages 308–322, 2005.

[BC76] A. Borodin and S. Cook. On the number of additions to compute specific polynomials. *SIAM J. on Computing*, 5:146–157, 1976.

[BC82] A. Borodin and S. Cook. Time-space tradeoffs for sorting on a general sequential model of computation. *SIAM J. on Computing*, 11:287–297, 1982.

[BCH69] Allan Borodin, Robert L. Constable, and John E. Hopcroft. Dense and non-dense families of complexity classe. In *FOCS*, pages 7–19, 1969.

[BDBK+94] S. Ben-David, A. Borodin, R. Karp, G. Tardos, and A. Wigderson. On the power of randomization in on-line algorithms. *Algorithmica*, 11:2–14, 1994.

[BEY98] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[BFadH+87] A. Borodin, F. Fich, F. Meyer auf der Heide, E. Upfal, and A. Wigderson. A time space tradeoff for element distinctness. *SICOMP*, 16(1):97–99, 1987.

[BFK+81] A. Borodin, M. Fischer, D. Kirkpatrick, N. Lynch, and M. Tompa. A time-space tradeoff for sorting on non oblivious machines. *JCSS*, 22(3):351–364, 1981.

[BH85] Allan Borodin and John E. Hopcroft. Routing, merging, and sorting on parallel models of computation. *J. Comput. Syst. Sci.*, 30:130–145, 1985.

[BIRS95] A. Borodin, S. Irani, P. Raghavan, and B. Schieber. Competitive paging with locality of reference. *JCSS*, 50(2):244–258, 1995.

[BKR$^+$01] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, and D. P. Williamson. Adversarial queuing theory. *JACM*, 48(1):13–38, 2001.

[BLS92] A. Borodin, N. Linial, and M. Saks. An optimal online algorithm for metrical task systems. *JACM*, 39(4):745–763, 1992.

[BM71] A. Borodin and I. Munro. Evaluating polynomials at many points. *Information Processing Letters*, 1(2):66–68, 1971.

[BM74] A. Borodin and R. Moenck. Fast modular transforms. *JCSS*, 8:366–386, 1974.

[BM75] A. Borodin and I. Munro. *Computational Complexity of Algebraic and Numeric Problems*. American Elsevier, 1975.

[BNR03] A. Borodin, M. N. Nielsen, and C. Rackoff. (Incremental) priority algorithms. *Algorithmica*, 37(4):295–326, 2003.

[Bor72] A. Borodin. Computational complexity and the existence of complexity gaps. *JACM*, 19(1):158–174, 1972.

[BOR03] A. Borodin, R. Ostrovsky, and Y. Rabani. Lower bounds for high dimensional nearest neighbor search and related problems. *Discrete and Computational Geometry – The Goodman-Polack FestschriftB. Aronov, S. Basu, J. Pach, M. Sharir, eds in the series: Algorithms and Combinatorics, Springer Verlag, Berlin*, 25:255–276, 2003.

[BvzGH82] Allan Borodin, Joachim von zur Gathen, and John E. Hopcroft. Fast parallel matrix and gcd computation. *Information and Control*, 52:241–256, 1982.