Proceedings of the International Congress of Mathematicians
Berkeley, California, USA, 1986

The Work of Leslie G. Valiant

V. STRASSEN

Theoretical computer science is very young, when compared for instance to number theory, geometry, or topology. While these classical fields are like magnificent old oaks, whose growth takes place at dizzying heights and is therefore not easy to follow, theoretical computer science resembles a fast-growing young tree, whose fresh green may be perceived and enjoyed by everyone coming near.

Leslie G. Valiant has contributed in a decisive way to the growth of almost every branch of this young tree. In order to convey some impression of the scope of his work and the impetuous pace of its creation, I shall first discuss three early papers, published within a single year, which contain spectacular advances in three very different areas. Then I will turn to what is perhaps Valiant's most important and mature work, centering around his theory of counting problems.

Languages. *Context-free grammars* were introduced by N. Chomsky in 1956 as a means of analyzing natural languages. They are now being used extensively for describing the structure of programming languages. The central algorithmic problem is to recognize sentences of the language defined by such a grammar. For a number of years recognition algorithms which run in a time proportional to n^3 on sentences of length n had been known, but in spite of much effort no significant improvement had been obtained except in special cases.

In a coup de main Valiant showed in 1975 that the recognition problem can be solved in less than cubic time by reducing it to integer matrix multiplication. He has never come back to this subject, but ingenious algorithmic reductions from combinatorial to algebraic problems have become one of the main themes of his work.

Graphs. Let m be a positive integer. An *m-superconcentrator* is a directed graph with m input and m output nodes, such that for every $r \leq m$ any r input nodes may be connected to any r output nodes in some order by r disjoint directed paths. By the size of an m-superconcentrator, one means its number of edges.

Superconcentrators first appeared in algebraic complexity theory: Any straightline algorithm for computing the Discrete Fourier Transform of order m

yields an m-superconcentrator of a size proportional to the length of the algorithm. In particular, the Fast Fourier Transform provides examples of m-superconcentrators of size const $\cdot m \log m$ and any improvement of the Fast Fourier Transform would lead to m-superconcentrators of still smaller size. Aho, Hopcroft, and Ullman, among others, stated the problem of proving or disproving that the minimal size of m-superconcentrators grows like $m \log m$. By the previous remarks a positive answer would yield an optimality proof for the Fast Fourier Transform up to order of magnitude.

Valiant dashed such hopes by showing that there exist m-superconcentrators of size linear in m. Considering the flexibility of these graphs the result is almost unbelievable. Valiant's proof, which is based upon previous work of Pinsker, combines a counting argument with an elegant recursive construction. (Let me note that Margulis and Gabber-Galil have succeeded in replacing the counting argument by an explicit construction as well.) In the decade after their discovery superconcentrators of linear size have become useful tools in the information and communication sciences far beyond their original purpose.

This work is just one example of Valiant's systematic and penetrating study of efficient imbedding and routing properties of graphs, leading in recent years to a theory of the general purpose parallel computer (the so-called supercomputer).

Turing machines. Since their invention by A. Turing in 1936 Turing machines have been the principal theoretical model on which notions of computability and computational complexity have been based. Often a Turing machine is used as a decision procedure for some property of numbers, graphs, logical formulas, etc., which by a suitable encoding are presented to the machine as binary strings. While recursion theory is concerned with the decidability of decision problems, complexity theory also considers the amount of time and space needed to reach a decision.

Given a function $t\colon \mathbf{N} \to \mathbf{N}$, let TIME$(t)$ be the class of all decision problems (i.e., sets of binary strings) that can be decided by a multitape Turing machine using only $O(t(n))$ computational steps on inputs of length n. Define SPACE(t) similarly in terms of the number of tape squares visited. Obviously TIME$(t) \subset$ SPACE(t), since in one step a Turing machine can reach at most a constant number of new tape squares.

The dualism of time and space, which to a large extent shapes the physical sciences, is also present in the discrete world of idealized computers. As a first but fundamental question we may ask whether the above inclusion is strict:

$$\mathrm{TIME}(t) \subsetneqq \mathrm{SPACE}(t)? \qquad \text{(A)}$$

Computing experience overwhelmingly indicates that this is indeed the case, but there does not seem to exist an easy proof. It was a scientific sensation when Hopcroft, Paul, and Valiant showed that in fact one has

$$\mathrm{TIME}(t) \subset \mathrm{SPACE}(t/\log t), \qquad \text{(B)}$$

which implies (A), since SPACE($t/\log t$) is strictly contained in SPACE(t) by a standard diagonalization argument. (Alluding to H. Weyl's classic the result might be stated: Space or time—it matters.)

Hopcroft, Paul, and Valiant reduced the proof of (B) to finding a good strategy for a certain game on graphs, the "pebble game," which had been invented earlier by Paterson and Hewitt for a different purpose. This strategy was later shown to be optimal by Paul, Tarjan, and Celoni, so that no improvement of (B) may be expected by the same method. (It is a pleasant coincidence that this optimality proof uses linear sized superconcentrators as an essential ingredient.)

The result of Hopcroft, Paul, and Valiant is weak in the following sense: The complexity classes TIME(t) and SPACE(t) depend on the Turing maching model of computation and therefore the validity of (A) or (B) may be lost by a change of models. A more robust formulation of the problem of time and space involves the complexity classes

$$\mathrm{PTIME} = \bigcup_k \mathrm{TIME}(n^k) \quad \text{and } \mathrm{PSPACE} = \bigcup_k \mathrm{SPACE}(n^k)$$

of all problems decidable in polynomial time or space. There is no doubt that the inclusion

$$\mathrm{PTIME} \subset \mathrm{PSPACE}$$

is also strict, but this has not yet been proven.

The class PTIME, or simply P, as it is called, has gained a central position in complexity theory, since it appears to be best suited for distinguishing between what can be and what cannot be computed in practice. For brevity I shall call the problems in P easy, those not in P hard.

Complete problems. Consider a map $f\colon \mathbf{N} \to 2^{\mathbf{N}}$ such that

$$\{(x,y)\colon y \in f(x)\} \quad \text{is easy,} \tag{1}$$

$$y \in f(x) \Rightarrow |y| \le |x|^k \tag{2}$$

for a suitable constant k. ($|x|$ denotes the binary length of x). The set of all numbers x such that $f(x)$ is nonempty is called a *search problem*. The complement of the set of primes is an example: $f(x)$ may be taken to consist of all proper divisors of x. The name "search problem" refers to the possibility of searching through all y satisfying the inequality in (2) for an element of $f(x)$. However, although each test of membership is easy by (1), such an exhaustive search may use exponential time due to the number of tests to be conducted.

The class NP of all search problems lies between P and PSPACE. It turns out that a great number of decision problems occurring in mathematics and its applications belong to NP, when suitably encoded. Here are a few examples: to decide

if a propositional formula is satisfiable,
if a graph is isomorphic to a subgraph of another,
if a graph has a Hamilton circuit,

if a graph has a 3-coloring,

if a diophantine equation of the form $F(X_1, \dots, X_n) = c$, where c and the coefficients of F are natural numbers, has a solution in natural numbers.

In his seminal paper, "The Complexity of Theorem Proving Procedures," S. Cook in 1971 showed that the satisfiability problem as well as the subgraph problem are *complete* in the class NP under polynomial reduction, in short, NP-complete. Roughly speaking, NP-complete problems have maximal degree of difficulty among all search problems. It follows that if either the satisfiability or the subgraph problem is easy, then every search problem is easy.

A more precise formulation of Cook's theorem implies an even stronger statement; namely, that a fast algorithm for deciding the satisfiability or the subgraph problem would create a fast decision procedure for any effectively given search problem in a completely mechanical way. Thus it could be used as a masterkey for search problems from all branches of mathematics. For instance, no additional competence in number theory would be needed for designing a fast test of primality or of representability of a number by any given positive polynomial. This appears so unlikely that there is little doubt about the validity of what we call *Cook's hypothesis*; namely, that the satisfiability and the subgraph problems are indeed hard, or equivalently that

$$\mathrm{P} \neq \mathrm{NP}.$$

The list of problems proved NP-complete was significantly extended by R. Karp in 1972 to include among others the Hamilton circuit and the graph coloring problems above. By now, most of the naturally occurring search problems have been classified as either easy or NP-complete. (The solvability of positive diophantine equations is NP-complete, even if it is restricted to binary quadratic polynomials, as has been shown by Manders and Adleman.)

Let me turn to Valiant's work on the subject. Given a map $f\colon \mathrm{N} \to 2^{\mathrm{N}}$ as in the definition of a search problem, we may not only ask whether $f(x)$ is nonempty, but may inquire about its size. Valiant calls the function $x \mapsto \# f(x)$ a *counting problem* and shows that the class of all counting problems also contains complete members, for instance, the counting problems corresponding to the NP-complete search problems of our list. Since counting solutions is at least as difficult as deciding whether a solution exists, complete counting problems are hard under Cook's hypothesis. Most exciting is Valiant's discovery in 1979 of various complete counting problems that correspond to easy search problems, thereby considerably enlarging the scope of the theory of NP-completeness. I give three examples:

(I) Counting subtrees of a directed graph. Note that if "subtrees" is replaced by "spanning subtrees," the counting problem becomes easy in view of a variant of a classical theorem of Kirchhoff (1847).

(II) Evaluating the probability of failure of an unreliable connecting network. According to Laplace's definition of probability as a proportion, this is tantamount to a counting problem.

(III) Counting perfect matchings of a bipartite graph. Here the corresponding search problem is not trivial as in I and II, but it is easy by a well-known algorithm of M. Hall.
In Valiant's treatment III is the critical example. The proof of completeness of this problem intricately combines ideas belonging to mathematical logic, graph theory, and algebra.

The number of perfect matchings of a bipartite graph is equal to the value of the permanent function at the zero-one-matrix representing the graph—over the ring of integers. What happens if we replace $\mathbf{Z}$ by $\mathbf{Z}/m\mathbf{Z}$? When $m = 2$ the permanent coincides with the determinant and is therefore easy to compute. When m is any fixed power of 2, Valiant shows that the problem remains easy. In contrast, he obtains the wonderful result that the existence of a fast algorithm for the permanent modulo m for some m that is not a power of 2 implies that any polynomially bounded number-theoretical function, whose graph is easy to decide, is itself easy to compute. ("Polynomially bounded" is to be understood in terms of lengths.) It can be deduced that if the permanent modulo 3 (say) is easy, then so is prime factorization of integers.

The permanent is a polynomial function of the entries in the matrix. Thus Valiant was naturally led into *algebraic complexity* theory, which we have already touched upon when discussing superconcentrators. Here the basic model is that of a straightline algorithm, i.e., a finite sequence of arithmetical instructions, to be executed over a suitable algebraic structure. For a number of years it had seemed that this subject would remain unaffected by the notion of NP-completeness. Motivated by his work on the permanent, however, Valiant developed a convincing analogue of the theory of search and counting problems entirely in the algebraic framework. The new theory differs considerably from its model, and it will not be possible to describe even its main features here. However, the counterpart of Cook's hypothesis—let us call it Valiant's hypothesis—may be sandwiched between two succinct statements of a classical algebraic flavor: Fix a field F of characteristic $\neq 2$ and let $t(n)$ be the smallest number r such that the permanent of size n may be obtained from the determinant of size r by a simple substitution, i.e., one in which variables may be replaced only by variables or elements of F. Then Valiant's hypothesis implies that $t(n)$ grows faster than any power of n. In turn, if $t(n)$ grows even faster than $e^{(\log n)^q}$ for some q (for instance, if it grows exponentially), Valiant's hypothesis is true over the field F.

For some of you it may seem that the theories discussed here rest on weak foundations. They do not. The evidence in favor of Cook's and Valiant's hypotheses is so overwhelming, and the consequences of their failure are so grotesque, that their status may perhaps be compared to that of physical laws rather than that of ordinary mathematical conjectures. Nevertheless a traditional proof would be of great interest, and it seems to me that Valiant's hypothesis may be easier to confirm than Cook's (for example, by using the powerful methods of algebraic geometry).

The selection of Valiant's works that I have presented to you does not do justice to many of his other achievements of comparable importance: dealing with boolean complexity, probabilistic algorithms, monotone and parallel computation, artificial intelligence. They all give ample evidence of his astuteness, originality, and taste.

Theoretical computer science is in the stage of formulating its central problems and devising the proof techniques for their solution. Valiant has been eminently involved in this process, not only by answering a number of recalcitrant open questions, but above all by developing important new concepts, which have led him to discover deep and beautiful connections between problems that had seemed to be totally unrelated.

In every scientific discipline, finding fruitful concepts is a most demanding task. This is especially true for a new field, since there exist so many conceptual possibilities. But the rewards balance the difficulties: the young shoots of the sapling will become the main boughs of the full grown tree.

I have no doubt that this applies to the work of Leslie Valiant. Let me wish him, and the three Fields medalists, futures as bright as their scientific pasts.

SELECTED PUBLICATIONS OF L. G. VALIANT

1. *The equivalance problem for deterministic finite-turn pushdown automata*, Inform. and Control **25** (1974), 123–133.

2. *Parallelism in comparison problems*, SIAM J. Comput. **4** (1975), 348–355.

3. *General context-free recognition in less than cubic time*, J. Comput. System Sci. **10** (1975), 308–315.

4. *On non-linear lower bounds in computational complexity*, Seventh Annual ACM Symposium on Theory of Computing (Albuquerque, N.M., 1975), Assoc. Comput. Mach., New York, 1975, pp. 45–53. (See also J. Comput. System Sci. **13** (1976), 278–285.)

5. *On time versus space and related problems* (with J. E. Hopcroft and W. J. Paul), 16th Annual Symposium on Foundations of Computer Science (Berkeley, Calif., 1975), IEEE Computer Society, Long Beach, Calif., 1975, pp. 57–64. (See also J. Assoc. Comput. Mach. **24** (1977), 332–337.)

6. *Graph-theoretic arguments in low-level complexity*, Lecture Notes in Comput. Sci., vol. 53, Springer-Verlag, 1977, pp. 162–176.

7. *Fast probabilistic algorithms for Hamiltonian circuits and matchings* (with D. Angluin), J. Comput. System Sci. **18** (1979), 155–193.

8. *The complexity of computing the permanent*, Theoret. Comput. Sci. **8** (1979), 189–201.

9. *The complexity of enumeration and reliability problems*, SIAM J. Comput. **8** (1979), 410–421.

10. *Completeness classes in algebra*, Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing (Atlanta, Ga., 1979), Assoc. Comput. Mach., New York, 1979, pp. 249–261.

11. *Negation can be exponentially powerful*, Theoret. Comput. Sci. **12** (1980), 303–314.

12. *Reducibility by algebraic projections*, Logic and Algorithms (Zurich, 1980), Monograph. Enseign. Math., No. 30, Univ. Genève, Geneva, 1982, pp. 365–380.

13. *A scheme for fast parallel communication*, SIAM J. Comput. **11** (1982), 350–361.

14. *Universal schemes for parallel communication* (with G. J. Brebner), Proceedings of the 13th ACM Symposium on Theory of Computing (Milwaukee, Wisconsin, 1981), Assoc. Comput. Mach., New York, pp. 263–277.

15. *Fast parallel computation of polynomials using few processors* (with S. Skyum, S. Berkowitz, and C. Rackoff), SIAM J. Comput. **12** (1983), 641–644.

16. *Short monotone formula for the majority function*, J. Algorithms **5** (1984), 363–366.

17. *A theory of the learnable*, Comm. ACM **27** (1984), 1134–1142.

18. *Learning disjunctions of conjunctions*, Proceedings of Ninth International Joint Conference on Artificial Intelligence (Los Angeles, Calif., 1985), pp. 560–566.

19. *A complexity theory based on Boolean algebra* (with S. Skyum), J. Assoc. Comput. Mach. **32** (1985), 484–502.

20. *A logarithmic time sort for linear size networks* (with J. H. Reif), J. Assoc. Comput. Mach. **34** (1987), 60–76.

21. NP *is as easy as detecting unique solutions* (with V. V. Vazirani), Theoret. Comput. Sci. (to appear).

INSTITUT FÜR ANGEWANDTE MATHEMATIK DER UNIVERSITÄT ZÜRICH, 8032 ZÜRICH, SWITZERLAND