The Structural Complexity Column
by
Juris HARTMANIS
Cornell University, Department of Computer Science
Ithaca, NY 14853, USA

RECENT PROGRESS IN INFORMATION–BASED COMPLEXITY

J. F. Traub[1] H. Woźniakowski[1,2]

[1]Department of Computer Science
Columbia University
New York, New York 10027
[2]Institute of Applied Mathematics
University of Warsaw
Banacha 2, 02-097 Warsaw, Poland

October 28, 1993

Table of Contents

The research reported here was supported in part by the National Science Foundation and the Air Force Office of Scientific Research. The article was written while the authors were visitors at the International Computer Science Institute, Berkeley, California .

1. Overview of Information-Based Complexity

The goal of this article is to report some of the recent progress in information-based complexity, which for brevity will be denoted as IBC. We have selected topics which might be of particular interest to the EATCS audience. We take an informal approach in this article, focusing mainly on ideas. For precise formulations and results, as well as proof techniques, see the books TW[1][80], TWW [83], Novak [88], TWW [88], Werschulz [91], and recent surveys , PT [87], PW [87], TW [91a, 91b], Heinrich [92], and Novak [93].

We begin by presenting a greatly simplified picture of computational complexity to indicate where IBC fits in. For our present purpose, computational complexity may be divided into two branches, discrete and continuous. Continuous computational complexity may again be split into two branches. The first, which we'll call *continuous combinatorial complexity*, deals with problems for which the information is *complete*. Problems where the information may be complete are those which are specified by a finite number of parameters. Examples include linear algebraic systems, matrix multiplication, and systems of polynomial equations. Blum, Shub and Smale [89] obtained the first NP-completeness results over the reals for a problem with complete information.

The other branch of continuous computational complexity is IBC. Typically, IBC studies infinite-dimensional problems. These are problems where either the input or the output are elements of infinite-dimensional spaces. Since digital computers can handle only finite sets of numbers, infinite-dimensional objects such as functions on the reals must be replaced by finite sets of numbers. Thus, complete information is not available about such objects. Only *partial* information is available when solving an infinite-dimensional problem on a digital computer. Typically, information is *contaminated* with errors such as round-off error, measurement error, and human error. Thus, the available information is partial and/or contaminated.

We want to emphasize this point for it is central to IBC. *Since only partial and/or contaminated information is available, we can solve the original problem only approximately. A goal of IBC is to obtain the computational complexity of computing such an approximation.*

In Figure 1 we schematized the structure of computational complexity described above.

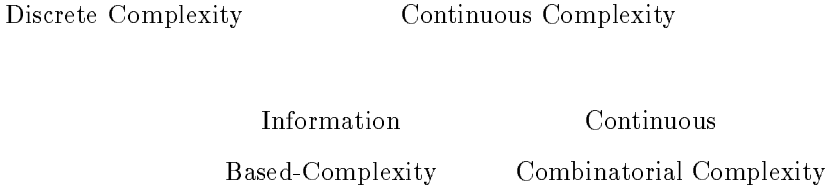

Figure 1

The motivation for studying IBC is two-fold:

(1) Continuous models, typically infinite-dimensional, are very common in science, engineering, economics, and even in finance. Examples of the mathematical problems which arise from these models are partial or ordinary differential equations, multivariate integration, and optimization.

[1]When one of us is a co-author, the citation will be made using only initials

(2) The subject matter covered by IBC is rich from a complexity point of view with many results and numerous open questions, as we hope to illustrate in this article.

Although IBC typically studies infinite-dimensional problems there are important exceptions. These include probabilistic complexity of processor synchronization with stochastic delays, Wasilkowski [88a], and complexity of solving large linear systems, TW [84], Nemirovsky [91, 92].

IBC is formulated as an abstract theory; see the Appendix. The applications often involve multivariate functions over the reals. For example, in multivariate integration, the integrand is a multivariate function. In optimization, one seeks an extremum of a multivariate function subject to multivariate constraints. In an initial-value problem, such as the wave equation, the initial condition is again specified by a multivariate function.

The observation that a function over the reals cannot be entered into a digital computer lies at the heart of IBC. (In the general case, an element of an abstract space cannot be entered.) We call a multivariate function *a mathematical input*, denoted by $\mathrm{I}_{\mathrm{math}}$. Let S be a linear or nonlinear operator which specifies the problem we want to solve, $S : F \to G$ for some sets F and G. The operator S carries $\mathrm{I}_{\mathrm{math}}$ from F into a mathematical output $\mathrm{O}_{\mathrm{math}}$ in G; see Figure 2(a)

$\mathrm{I}_{\mathrm{math}}$ S $\mathrm{O}_{\mathrm{math}}$

Figure 2(a)

Of course, this is too general to characterize an IBC problem. For example, $\mathrm{I}_{\mathrm{math}}$ could be the locations of a set of cities and $\mathrm{O}_{\mathrm{math}}$ could be an optimal tour; which is a typical discrete problem. This is an IBC problem when $\mathrm{I}_{\mathrm{math}}$ cannot be entered into a digital computer, and it must be replaced by a computer input denoted by $\mathrm{I}_{\mathrm{comp}}$.

The computer input, $\mathrm{I}_{\mathrm{comp}}$, consists of a finite set of numbers. For example, if $\mathrm{I}_{\mathrm{math}}$ is a function then $\mathrm{I}_{\mathrm{comp}}$ might consist of its values at certain points. $\mathrm{I}_{\mathrm{comp}}$ is obtained from $\mathrm{I}_{\mathrm{math}}$ by *information operations*. Different disciplines have different names for these information operations. Computer scientists called them oracle calls, mathematicians call them functionals, and engineers call them black-box calls. The replacement of $\mathrm{I}_{\mathrm{math}}$ by $\mathrm{I}_{\mathrm{comp}}$ may be viewed as a discretization.

Denote the set of information operations by $N(\mathrm{I}_{\mathrm{math}})$; we call N the *information* operator. Since many (typically, an infinite number of) mathematical inputs map into the same computer input, the mapping N is many-to-one. That is, discretization is irreversible. The situation is diagrammed in Figure 2(b).

$\mathrm{I}_{\mathrm{math}}$ S $\mathrm{O}_{\mathrm{math}}$

$\mathrm{I}_{\mathrm{comp}}$

Figure 2(b)

Although there has been mention of neither computer output nor algorithm, we can already draw certain conclusions. Since N is a many-to-one map, the computer does not know the mathematical input. Therefore, it is impossible to solve this problem exactly; the best we can hope for is an approximation.

We assign the same cost to each information operation. Given an error threshold ε, we can define the information complexity, $\mathrm{COMP}^{\mathrm{info}}(\varepsilon)$, as the minimal cost of the information operations needed to obtain an ε-approximation. (In computational learning theory this is called sample complexity.) Information complexity can be defined in different settings such as the worst case, average case or probabilistic setting.

Using the concept of *radius of information*, $r(N)$, see TW [80, pp. 9-15], TWW [88, pp. 43-45, 197-200, 327-328], we can often obtain sharp lower and upper bounds on the information needed to get an ε-approximation. The information N is powerful enough to obtain an ε-approximation iff

$$r(N) \le \varepsilon.$$

Since the information complexity is a lower bound on the computational complexity, defined below, this has led to *proven* (not conjectured) intractability and unsolvability results which we'll describe in Section 2.

Because of the basic role played by information-level results we decided to name this area information-based complexity. This level typically does not exist for discrete problems. However, combinatorial issues will play an increasingly important role in IBC; see Section 4.

Let the computer output be denoted by $\mathrm{O}_{\mathrm{comp}}$ and the operator that maps $\mathrm{I}_{\mathrm{comp}}$ into $\mathrm{O}_{\mathrm{comp}}$ by ϕ. We call ϕ a combinatory algorithm (algorithm for brevity). Since ϕ maps the computer input into the computer output it plays the same role as algorithm does elsewhere in computer science. Figure 2(c) completes the picture.

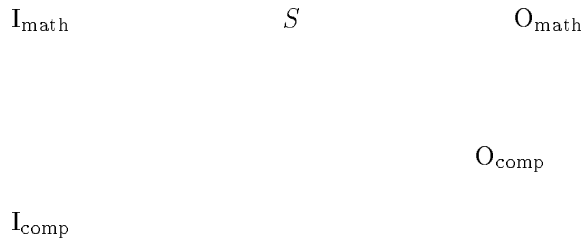

Figure 2(c)

Observe that $\mathrm{O}_{\mathrm{comp}} \neq \mathrm{O}_{\mathrm{math}}$ because N is many-to-one. In other words, S does not commute with ϕ composed with N.

We now discuss the model of computation used in IBC. For simplicity, we restrict ourselves to the case that $G = \mathbb{R}$. We assume that the real number model is chosen as our model of computation. (See Section 5 for a discussion of why the real number model is often used in IBC and also of research on finite models.) That is, we assume that arithmetic operations and comparisons on real numbers are carried out exactly and at unit cost.

We define the combinatorial complexity, $\mathrm{COMP}^{\mathrm{comb}}(\varepsilon)$, as the minimal cost of the combinatory operations needed to compute an ε-approximation if all information operations were free.

Finally, we define the computational complexity, $\mathrm{COMP}(\varepsilon)$, as the minimal cost of computing the computer output with error at most ε under the assumption that information and combinatory operations are charged.

As before, combinatorial and computational complexity may be defined in the worst case, average case and probabilistic settings. Note that,

$$\mathrm{COMP}(\varepsilon) \ge \max\{\mathrm{COMP}^{\mathrm{info}}(\varepsilon), \mathrm{COMP}^{\mathrm{comb}}(\varepsilon)\}.$$

We conclude this overview by characterizing IBC and stating its major goals. IBC studies problems which have the properties listed below.

(1) $\mathrm{I}_{\mathrm{comp}} \neq \mathrm{I}_{\mathrm{math}}$.

(2) There is a charge for obtaining $\mathrm{I}_{\mathrm{comp}}$.

We discuss the first of these. These are two major reasons why $\mathrm{I}_{\mathrm{comp}} \neq \mathrm{I}_{\mathrm{math}}$. The first is that the mathematical input cannot be represented by a finite set of numbers. We say the information about $\mathrm{I}_{\mathrm{math}}$ is *partial*. An important example in applications is when $\mathrm{I}_{\mathrm{math}}$ is a multivariate function. A second reason is that the information about $\mathrm{I}_{\mathrm{math}}$ is *contaminated*. Information may be contaminated because of round-off or measurement errors.

We list some of the major goals of IBC.

(1) Obtain good lower and upper bounds on the computational complexity, information complexity, and combinatorial complexity.

(2) Find information N and an algorithm ϕ for which the computational complexity is attained or nearly attained. Such N and ϕ are called *optimal*, or *nearly optimal*.

We summarize the reminder of this article. We will present a selection of recent results from a number of IBC areas. We then conclude this article with a discussion of similarities and differences with discrete complexity and a brief history. An abstract formulation of IBC may be found in the Appendix.

2. Breaking Intractability

It has been established that in the worst case deterministic setting many problems studied in IBC are unsolvable or intractable. More precisely, let the mathematical input f be a multivariate function of d variables. Let the smoothness of the set of inputs be denoted by r. For example, we might require that all partial derivatives of f up to order r exist and are uniformly bounded by 1. Assume we want to guarantee an error at most ε. Then, for many continuous problems the worst case computational complexity, $\mathrm{COMP}(\varepsilon)$, is given by

$$\mathrm{COMP}(\varepsilon) \;=\; \Theta\left(\left(\frac{1}{\varepsilon}\right)^{d/r}\right). \tag{1}$$

For example, multivariate integration, function approximation, partial differential equations, integral equations, and nonlinear optimization all have this computational complexity, see Bakhvalov [59], Heinrich [93], Nemirovsky and Yudin [83], Novak [88], Pereverzev [89], TWW [88], and Werschulz [91].

Furthermore, many problems in science, engineering, economics and even finance use mathematical models with large d. For example, computational chemistry, computational design of pharmaceuticals, and computational metallurgy involve computation with large number of particles. Since the specification of each particle requires three variables for static problems and six variables for dynamic problems, this leads to problems with very large d. For path integrals, important in the foundation of physics, $d = +\infty$; they invite approximation by multivariate integration with huge d. Problems with large d are also important in mathematical disciplines such statistics and geometry.

Observe that we can conclude that if the smoothness r is fixed and positive then the computational complexity is an exponential function in d. Thus, problems whose complexity is governed by (1) are intractable in d. If $r = 0$, that is, if the class of inputs is only continuous, then $\mathrm{COMP}(\varepsilon) = +\infty$ for small ε; that is,

the problem is *unsolvable*.

The only way to break unsolvability or intractability is to weaken the assurance of an ε-approximation by shifting to another setting. Three settings have been used for trying to break intractability: randomized, average case, and probabilistic settings. Here we confine ourselves to recent advances on breaking intractability in the average case setting. See TW [91a] for a survey of how to break intractability in the randomized setting.

We describe recent advances in breaking intractability for multivariate integration and multivariate function approximation. Multivariate integration is especially common since computing the expectation of any stochastic process leads to this problem.

In the average case setting the average computational complexity, $\mathrm{COMP}^{\mathrm{avg}}(\varepsilon)$, is defined as the minimal expected cost such that the average error is less than ε. One has to put a measure on the space of inputs. Although for discrete problems one can assume that all inputs are equiprobable, no such assumption can be made for typical sets of functions. The most commonly used measures on function spaces are Gaussian measures, and, in particular, Wiener measures which are a special kind of Gaussian measure.

It was known that multivariate integration is tractable on the average but the proof is non-constructive. That is, the optimal points at which the integrand should be evaluated and the average computational complexity were unknown.

Then W [91] established a relation between discrepancy and the average complexity of multivariate integration. Discrepancy has been extensively studied in number theory and sharp bounds on discrepancy in d dimensions were established by Roth [54,80]. The use of the results from discrepancy theory solved the multivariate integration problem.

We describe the results more precisely. Let $r = 0$. Recall that in the worst case deterministic setting the problem is unsolvable. Assume the measure on the integrands is the Wiener sheet measure. Then

$$\mathrm{COMP}^{\mathrm{avg}}(\varepsilon) \;=\; \Theta\left(\frac{1}{\varepsilon}\left(\log\frac{1}{\varepsilon}\right)^{(d-1)/2}\right).$$

Thus a problem which is worst-case unsolvable becomes tractable on the average.[2] Either Hammersley points or hyperbolic-cross points are nearly optimal as the evaluation points in d dimensions. These results were generalized to the case of smooth inputs by Paskov [93].

We turn to the average complexity of function approximation. This is particularly important since unlike for multivariate integration, it is known that randomization does not help for function approximation, see Wasilkowski [88b], Novak [92]. Again, let $r = 0$ and assume a Wiener sheet measure. Then

$$\mathrm{COMP}^{\mathrm{avg}}(\varepsilon) \;=\; \Theta\left(\frac{1}{\varepsilon^2}\left(\log\frac{1}{\varepsilon}\right)^{2(d-1)}\right)$$

and again hyperbolic cross-points can be used; see W [92b].

Roth's discrepancy results and the average computational results quoted above are big theta results in ε. That is, the dependence on ε is known, but there is a multiplicative factor, $g(d)$, which is not known. If we're serious about solving problems with large d we must be able to bound $g(d)$. It is believed that obtaining good theoretical estimates of $g(d)$ is very hard.

[2]By tractable (in $1/\varepsilon$) we mean that the complexity is bounded by $K(d)\,(1/\varepsilon)^p$ for all d and $\varepsilon \le 1$ for a number p which is independent of d and ε.

The problem may be solved by getting rid of the factor $g(d)$ in the following way, W [93]. A problem is said to be *strongly tractable* if the number of information operations, $m(\varepsilon, d)$, needed to compute an ε-approximation is independent of d and depends polynomially on $1/\varepsilon$, that is,[3]

$$m(\varepsilon, d) \;\le\; K \left(\frac{1}{\varepsilon}\right)^p, \qquad \forall\, d,\ \forall\, \varepsilon \le 1,$$

for certain numbers K and p.

That might seem to much to expect but multivariate integration and multivariate approximation are both strongly tractable on the average[4] and it is sufficient to take the information operations as function evaluations, W [93]. Usually in computational complexity, an upper bound is given by an algorithm and a lower bound by a theorem. But in this case, the upper bound has been determined by a theorem and is *non-constructive*. That is, we know that there must exist sample points at which we should evaluate the function and a combinatory algorithm which make multivariate integration and approximation strongly tractable. The construction of such sample points and algorithm is being studied; WW [94].

Due to the relation between discrepancy and average case multivariate integration, strong tractability for multivariate integration implies that the discrepancy of n points in d dimensions can be bounded, independently of d, by $K\, n^{-p}$ with the same K and p for both problems. This estimate is of interest in its own right since discrepancy is of considerable interest in number theory, see Beck and Chen [87], and Niederreiter [92]. Furthermore there are numerous applications of discrepancy; for example, for applications in computer graphics, see Dobkin and Mitchell [93].

3. Verification

Most of IBC has been devoted to the computational complexity of computing an ε-approximation. Recently, the computational complexity of *verification* has been studied, that is checking whether an answer is correct, see W [92a]. In addition to being given a problem, we are also given an "answer" g and asked whether it is true that g is within ε of the mathematical output; see the Appendix for a precise definition.

The reader's reaction may be that, of course, verification is no harder than computation. Indeed, if the mathematical output can be computed exactly at finite cost, as is the case for discrete problems, then with one extra comparison one can solve the verification problem.

However, for typical IBC problems the mathematical output *cannot* be computed with finite cost, and the relation between verification and computation is not obvious. As we shall see, in the worst case setting verification may be unsolvable while the corresponding computational problem is easy.

We illustrate this with a simple example. The computational problem is to compute an ε-approximation to $\int_0^1 f(x)\,dx$ where the mathematical input f is an arbitrary function over $[0,1]$ satisfying a Lipschitz condition with constant at most one. The computational input is given by values of f at some points. The computational complexity in the worst case setting is known to be of order $1/\varepsilon$; thus the computational problem is "easy".

Suppose now that we're given the purported answer g and asked to check whether this is within ε of the integral of f. We show that the verification problem is unsolvable.

[3]More precisely, it is required that the computational complexity can be bounded by $K\, c(d)\, (1/\varepsilon)^p$ for certain numbers K and p, independent of d and ε, where $c(d)$ is the cost of one information evaluation of a function of d variables.

[4]We stress that this holds for the Wiener sheet measure. For an isotropic Wiener measure, function approximation is still intractable even on the average, see Wasilkowski [93].

Suppose that we compute f at a finite number of points x_i and that for every such point $f(x_i) = g + \varepsilon$. If we answer NO the adversary will choose $f(x) \equiv g + \varepsilon$. This function is certainly Lipschitz (with constant zero), and compatible with the computed function values. Since $\int_0^1 f(x)\,dx = g + \varepsilon$ is within ε of the answer g, we made a mistake by answering NO.

If we answer YES the adversary will choose a hat function f going through the points $(x_i, g + \varepsilon)$ and with Lipschitz constant one. Clearly, $\int_0^1 f(x)\,dx > g + \varepsilon$ which is *not* within ε of the answer g. We made a mistake by answering YES. Hence, as long as we have finitely many function values, there is no way to solve the verification problem in the worst case setting.

It can be shown that verification for IBC problems is often unsolvable in the worst case setting. Verification is therefore studied in the probabilistic setting. Here we want to verify that g is an ε-approximation with confidence δ; see the Appendix. In this setting the probabilistic complexity of verification depends on how ε and δ are related. Any relation between the probabilistic complexities of verification and computation is possible. In particular, verification can be exponentially (in δ) harder than computation.

NW [92] studied *relaxed* verification in the worst case setting. That is the answers can be YES, NO, or DON'T CARE. The size of the DON'T CARE region is specified by a parameter α; see the Appendix. For a positive α, the worst case complexity of relaxed verification is finite. It is related to the worst case complexity of the computational problem with ε replaced by roughly $\varepsilon\,\alpha^q$ with $q \in [0, 1]$ depending on the problem. Hence, if α is not too small, the complexity of relaxed verification is roughly comparable to the complexity of the computational problem. If, however, α is small then the complexity of relaxed verification is usually much larger than the complexity of the computational problem.

4. Combinatorial Complexity

To date, IBC problems have usually been proven unsolvable or intractable by showing that their *information complexity* was infinite or exponential. Recent results establish unsolvability or intractability by showing that the *combinatorial complexity* is infinite or, if P≠NP, not polynomial. We report these results and also pose an open question.

Papadimitriou and Tsitsiklis [86] is a pioneering paper which proves that a nonlinear problem in decentralized control theory is intractable if P≠NP. More precisely, the information complexity is a polynomial in $1/\varepsilon$ but the combinatorial complexity in a Turing machine model of computation is not polynomial in $1/\varepsilon$, if P≠NP.

WW [93] show that there exists a linear problem whose information complexity is a polynomial in $1/\varepsilon$ but whose combinatorial complexity is infinite[5], making the problem unsolvable. An "artificial" problem is constructed to show that even a linear problem can be very hard combinatorially. Chu [94] shows that the combinatorial complexity can be any increasing function of the information complexity.

We pose an open question. So far, tight bounds on the computational complexity of IBC problems are achieved when the minimal amount of information is used. Is there a problem for which more information operations should be used to achieve the computational complexity? That is, does there exist a problem for which the minimal amount of information is very hard to combine but if more information operations are computed then it is easier to combine them and the total cost of computing an ε-approximation is minimized

[5]This result holds if we allow arithmetic operations, comparisons of real numbers, and precomputation. It is open if there exists a linear problem with finite information complexity and infinite combinatorial complexity in the extended real number model in which logarithms, exponentials and ceilings are allowed.

in the latter case.

We believe that in the future, progress in IBC will increasingly require results in both information complexity and combinatorial complexity.

5. Similarities and Differences with Discrete Complexity

We begin with similarities. As in the rest of computational complexity, IBC studies lower and upper bounds on the computational difficulty of solving mathematically posed problems. Optimal and near-optimal algorithms are sought. To attempt to break the intractability results and conjectures of the worst case deterministic setting, both IBC and discrete complexity turned to other settings such as the randomized and average case settings.

There are also significant contrasts, three of which we will discuss in the remainder of the section. IBC has the following characteristics:

Problems cannot be exactly solved

Intractability has been proven for many problems

Real number model usually used

We discuss each of these.

Problems Cannot Be Exactly Solved

As discussed in Section 1, it is impossible to solve IBC problems exactly because $I_{comp} \neq I_{math}$. It is possible, in principle, to solve discrete problems exactly although one may choose to solve them approximately to reduce the cost.

Intractability has been proven for many problems

Using information-level arguments, unsolvability and intractability has been established for many IBC problems. With only a few exceptions, there are no non-trivial lower bounds on the combinatorial complexity of IBC problems. Since only combinatorial arguments are available, intractability of many discrete problems has been conjectured. (Of course, lower bounds, as well as unsolvability results, have been established for some combinatorial problems.)

Real number model usually used

To date, the real number model of computation has usually been used in continuous computational complexity. After discussing the motivation, we turn to finite models for continuous computational complexity.

Scientific problems are usually solved using fixed precision floating point arithmetic. The cost of floating point operations and comparisons is independent of the size of the operands. Furthermore, all arithmetic operations and comparisons cost about the same to execute. Our goal is to choose a model of computation that corresponds to performance of a digital computer executing floating point arithmetic. The abstraction we choose is the *real number model*, which assumes that arithmetic and comparisons on real numbers can be executed exactly and at unit cost. (The choice of unit cost is just scaling.) Rounding errors occur when a digital computer executes operations in fixed precision floating point arithmetic. In our abstraction we assume arithmetic is performed without error. This separation of complexity theory from error analysis is done for technical reasons; computational complexity theory is hard enough without including round-off error. When an interesting new algorithm is discovered from computational complexity considerations, a stability analysis in fixed precision floating point arithmetic must be performed.

We stress that the real number model is *not* polynomially equivalent to the Turing machine model. For example, TW [82] shows that the cost of Kachian's algorithm is not polynomial in the real number model and conjecture that linear programming is not polynomial in this model. This conjecture is still open.

Several finite models of computations have also been analyzed. One of them is a model based on recursive analysis, see Ko [91].

In the bit model it is assumed that one can get a rational binary approximation of a real number or of a function value to within any accuracy with the cost depending on the number of bits. This model has been studied for problems with *complete* information, for instance, for finding roots of polynomials, see Schönhage [86]. A mixed model, in which the bit model is used for information operations, and the real number model for combinatory operations, is utilized by Kacewicz and Plaskota [90] to analyze certain IBC problems.

It is, of course, desirable to fully explore finite models for IBC problems and we believe this to be an important direction for future research.

6. A Brief History

We present a very brief history of IBC. Research in the spirit of IBC was initiated in the Soviet Union by Kolmogorov in the late 40's. Nikolskij [50], a student of Kolmogorov, studied optimal quadrature. This line of research was greatly advanced by Bakhvalov, see e.g., Bakhvalov [59, 71]. In the United States research in the spirit of IBC was initiated by Sard [49] and Kiefer [53]. Kiefer reported the results of his 1948 MIT Master's Thesis that Fibonacci sampling is optimal when approximating the maximum of a unimodal function. Sard studied optimal quadrature.

Golomb and Weinberger [59] studied optimal approximation of linear functionals. Schoenberg [64] realized the close connection between splines and algorithms optimal in the sense of Sard.

T[61,64] initiated the study of iterative computational complexity, emphasizing the central role of information. Maximal order results, needed to obtain lower bounds on computational complexity, were obtained for scalar nonlinear equations. W [75] introduced the concept of order of information in an abstract space which provides a general tool for establishing maximal order of an algorithm.

Micchelli and Rivlin [77] studied optimal recovery and considered optimal error algorithms for the approximation of linear operators. Linear noisy information was permitted.

A general formulation of IBC, primarily in the worst case deterministic setting, is presented in TW [80], where a somehow more detailed history and an annotated bibliography of over 300 papers and books up to 1979 can be also found. At the time IBC was called analytic complexity to differentiate it from algebraic complexity. TWW [88] extend the study of IBC to numerous settings including average case, randomized, probabilistic, and asymptotic settings, as well as mixed settings.

Appendix

We present an abstract formulation of IBC. Let

$$S : F \to G$$

where F is a subset of a linear space and G is a normed linear space.

For $f \in F$, we wish to compute an approximation to $S(f)$. To do this we must know something about f. A basic assumption is that we have only partial information about f. We gather this partial information

about f by information operations $L(f)$. Here we will assume that L is a linear functional. Let Λ denote the class of information operations we will permit. The choice of Λ will depend on the problem we wish to solve. If we wish to approximate a definite integral we must exclude definite integration as a permissible information operation, and for this problem Λ is usually defined as the class of function evaluations. For other problems, such as the solution of nonlinear equations, we may permit any linear functional. Let

$$N(f) = [L_1(f), \ldots, L_n(f)],$$

for $L_i \in \Lambda$. Here L_i, as well as n, can be adaptively chosen depending on the already computed information operations.

$N(f)$ is called the *information* on f and N the *information operator*. The motivation for introducing the information operator N is to replace the element f, which is often from an infinite-dimensional space, by n numbers. An idealized algorithm[6] ϕ is an operator $\phi : N(F) \to G$. The approximation $U(f)$ is then computed by

$$U(f) = \phi(N(f)).$$

(The assumption that the approximation is the composition of ϕ with N is made without loss of generality.) We seek $U(f)$ such that

$$\|S(f) - U(f)\| \leq \varepsilon.$$

We say $U(f)$ is an *ε-approximation*.

We illustrate the abstract model by an integration example with

$$S(f) = \int_0^1 f(t)dt,$$

$$F = \{f : \ f \in C^r(0,1) \text{ and } \|f\|_{\max} \leq 1\},$$

and G as the set of real numbers. The functionals are chosen as $L_i(f) = f(t_i)$. An example of an algorithm is

$$U(f) \ = \ \phi(N(f)) = \frac{1}{n}\sum_{i=1}^{n} f(t_i).$$

To define computational complexity we must first introduce our model of computation, which is defined by two postulates:

(i) Let Ω denote the set of permissible combinatory operations including the addition of two elements in G, multiplication by a scalar in G, arithmetic operations on real numbers, and comparison of real numbers. We assume that each combinatory operation is performed exactly with unit cost.

(ii) We assume that we are charged for each information operation. That is, for every $L \in \Lambda$ and $f \in F$, the computation of $L(f)$ costs c, where $c > 0$. Typically, $c \gg 1$.

We assume the *real number model*, that is, we can perform operations on real numbers exactly and at unit cost. See Section 5 for a discussion and motivations underlying the model of computation and the real number model.

We briefly describe how the computation is carried out and how its cost is calculated. Let $\text{cost}(N, f)$ denote the cost of computing the information $N(f)$. Knowing the information $N(f)$, the approximation

[6]By using such a general definition of algorithm, we strengthen the lower bound conclusions. For upper bounds, we restrict the algorithms to those constructed from permissible combinatory operations.

$U(f) = \phi(N(f))$ is computed by combining the information to produce an element of G which approximates $S(f)$.

Let $\mathrm{cost}(\phi, N(f))$ denote the cost of computing $U(f) = \phi(N(f))$, given $N(f)$. Then the total cost of computing $U(f)$, $\mathrm{cost}(U, f)$, is

$$\mathrm{cost}(U, f) = \mathrm{cost}(N, f) + \mathrm{cost}(\phi, N(f)).$$

We are ready to define the computational complexity, $\mathrm{comp}(\varepsilon)$, as

$$\mathrm{comp}(\varepsilon) = \inf\{\mathrm{cost}(U) : U \text{ such that } e(U) \le \varepsilon\},$$

with the convention that $\inf \emptyset = \infty$. The definition of $\mathrm{cost}(U)$ and $e(U)$ varies according to the setting. Settings studied in IBC include worst case, average case, probabilistic, randomized and asymptotic. Mixed settings are also studied. We confine ourselves here to the definition of just the worst case and average case settings.

Worst Case Setting: The *worst case error* and *worst case cost* of U are defined by

$$e(U) = \sup_{f \in F} \|S(f) - U(f)\|,$$
$$\mathrm{cost}(U) = \sup_{f \in F} \mathrm{cost}(U, f).$$

Average Case Setting: Let μ be a probability measure defined on F. The *average case error* and *average case cost* of U are defined by

$$e(U) = \sqrt{\int_F \|S(f) - U(f)\|^2 \mu(df)},$$
$$\mathrm{cost}(U) = \int_F \mathrm{cost}(U, f)\mu(df).$$

The concept of complexity permits us to introduce the fundamental concepts of optimal information and optimal algorithm. Information N and an algorithm ϕ that uses N are called *optimal information* and *optimal algorithm*, respectively, iff $U = \phi \cdot N$ satisfies $\mathrm{cost}(U) = \mathrm{comp}(\varepsilon)$ and $e(U) \le \varepsilon$.

We define the verification problem. For given $g \in G$ we want to check whether $\|S(f) - g\| \le \varepsilon$. That is, we define $\mathrm{VER}(f, g)$ =YES if $\|S(f) - g\| \le \varepsilon$, and $\mathrm{VER}(f, g)$ =NO otherwise. In the worst case setting, we wish to find an approximation operator U such that

$$U(f, g) = \mathrm{VER}(f, g) \qquad \forall f \in F,\ g \in G.$$

In the probabilistic setting, we assume that the set F is equipped with a probability measure μ. For a given confidence parameter $\delta \in [0, 1]$, we wish to find an approximation operator U such that

$$\mu\{f \in F;\ U(f, g) = \mathrm{VER}(f, g)\} \ge 1 - \delta, \qquad \forall g \in G.$$

For relaxed verification, we assume that $\alpha \in [0, 1]$ and we redefine $\mathrm{VER}(f, g)$ as follows. We set $\mathrm{VER}(f, g)$ =YES if $\|S(f) - g\| \le \varepsilon$, $\mathrm{VER}(f, g)$ =NO if $\|S(f - g\| > (1 + \alpha)\varepsilon$, and $\mathrm{VER}(f, g)$ =DON'T CARE, otherwise.

The complexity of verification or relaxed verification is defined similarly as for computational problems, that is, by minimizing the cost of computing U that solves the corresponding verification problem.

Acknowledgments

We thank Z. Galil, G. W. Wasilkowski and A. G. Werschulz for valuable comments.

References

Bakhvalov, N. S. [59], *On approximate calculation of integrals (in Russian)*, Vestnik MGV, Ser. Mat. Mekh. Astron. Fiz. Khim. **4** (1959), 3–18.

Bakhvalov, N. S. [71], *On the optimality of linear methods for operator approximation in convex classes of functions (in Russian)*, Zh. Vychisl. Mat. Mat. Fiz. **11** (1971), 1014–1018 English transl. USSR Comput. Math. Math. Phys., 11, 1971, 244–249.

Beck, J. and Chen, W. W. L. [87], *Irregularities of Distribution*, Cambridge University Press, 1987.

Blum, L., Shub, M. and Smale, S. [89], *On a Theory of Computation and Complexity over the Real Numbers: NP-Complete- ness, Recursive Functions and Universal Machines*, Bull. Amer. Math. Soc. **21** (1989), 1–46.

Chu, M. [94], *There exists a problem whose computational complexity is any given function of the information complexity*, to appear in J. Complexity (1994).

Dobkin, D. P. and Mitchell, D. P. [93], *Random-Edge Discrepancy of Supersampling Patterns*, Graphics Interface '93, York, Ontario (1993).

Golomb, M. and Weinberger, H. F. [59], *Optimal approximation and error bounds*, in, *On Numerical Approximation (R. E. Langer, ed.)*, Univ. of Wisconsin Press, Madison, 1959, pp. 117–190.

Heinrich, S. [92], *Random Approximation in Numerical Analysis*, Proc. of the Functional Analysis Conference, Essen 1991, "Lecture Notes in Pure and Applied Mathematics", Marcel Dekker (1992).

Heinrich, S. [93], *Complexity of integral equations and relations to s-numbers*, J. Complexity **9** (1993), 141–153.

Kacewicz, B. Z. and Plaskota, L. [90], *On the minimal cost of approximating linear problems based on information with deterministic noise*, Numer. Funct. Anal. and Optimiz. **11** (1990), 511–528.

Kiefer, J. [53], *Sequential minimax search for a maximum*, Proc. Amer. Math. Soc. **4** (1953), 502–505.

Ker-I Ko [91], *Complexity Theory of Real Functions*, Birkhäser, Boston, Mass., 1991.

Micchelli, C. A. and Rivlin, T. J. [77], *A survey of optimal recovery*, in Optimal Estimation in Approximation Theory (C. A. Micchelli and T. J. Rivlin, eds.), Plenum, New York (1977).

Nemirovsky, A. S. [91], *On optimality of Krylov's information when solving linear operator equations*, J. Complexity **7** (1991), 121–130.

Nemirovsky, A. S. [92], *Information-Based Complexity of Linear Operator Equations*, J. Complexity **8** (1992), 153–175.

Nemirovsky, A. S. and Yudin, D. B. [83], *Problem Complexity and Method Efficiency in Optimization*, Wiley-Interscience, New York, 1983.

Niederreiter, H. [92], *Random Number Generation and Quasi-Monte Carlo Methods*, SIAM, Philadelphia, Penn., 1992.

Nikolskij, S. M. [50], *On the problem of approximation estimate by quadrature formulas (in Russian)*,, Usp. Mat. Nauk **5** (1950), 165–177.

Novak, E. [88], *Deterministic and Stochastic Error Bounds in Numerical Analysis*, Lectures Notes in Math. Springer Verlag, Berlin, **1349**, 1988.

Novak, E. [92], *Optimal linear randomization methods for linear operators in Hilbert spaces*, J. Complexity **8** (1992), 22–36.

Novak, E. [93], *Algorithms and complexity for continuous problems*, to appear in "Geometry, Analysis, and Mechanics" J. M. Rassias, ed., World Scientific, Singapore (1993).

Novak, E. and Woźniakowski, H. [92], *Relaxed Verification for Continuous Problems*, J. Complexity **8** (1992), 124–152.

Packel, E. W. and Traub, J. F. [87], *Information-based complexity*, Nature **328** (1987), 29–33.

Packel, E. W. and Woźniakowski, H. [87], *Recent developments in information-based complexity*, Bull. Amer. Math. Soc. **17** (1987), 9–36.

Papadimitriou, C. H. and Tsitsiklis, J. [86], *Intractable problems in control theory*, SIAM J. Control Optim. **24** (1986), 639–654.

Paskov, S. H. [93], *Average Case Complexity of Multivariate Integration for Smooth Functions*, J. Complexity **9** (1993), 291–312.

Pereverzev, S. V. [89], *On the complexity of the problem of finding solutions of Fredholm equations of the second kind with differentiable kernels (in Russian)*, Ukrain. Mat. Sh. **41** (1989), 1422–1425.

Roth, K. F. [54], *On irregularities of distribution*, Mathematika **1** (1954), 73–79.

Roth, K. F. [80], *On irregularities of distribution, IV*, Acta Arith. **37** (1980), 67–75.

Sard, A. [49], *Best approximate integration formulas; best approximation formulas*, Amer. J. Math. **71** (1949), 80–91.

Schönhage, A. [86], *Equation solving in terms of computational complexity*, Proc. Intern. Congress Math., Berkeley (1986).

Schoenberg, I. J. [64], *Spline interpolation and best quadrature formulas*, Bull. Amer. Math. Soc. **70** (1964), 143–148.

Traub, J, F. [61], *On functional iteration and the calculation of roots*, Preprints of papers, 16th Natl. ACM Conf. Session 5A-1, Los Angeles, Ca. (1961), 1-4.

Traub, J, F. [64], *Iterative Methods for the Solution of Equations*, Englewood Cliffs, N. J., 1964. Reissued: Chelsea Press, New York, 1982.

Traub, J. F., Wasilkowski, G. W. and Woźniakowski, H. [83], *Information, Uncertainty, Complexity*, Addison-Wesley, Reading, Mass., 1983.

Traub, J. F., Wasilkowski, G. W. and Woźniakowski, H. [88], *Information-Based Complexity*, Academic Press, New York, 1988.

Traub, J. F. and Woźniakowski, H. [80], *A General Theory of Optimal Algorithms*, Academic Press, New York, 1980.

Traub, J. F. and Woźniakowski, H. [82], *Complexity of linear programming*, Oper. Res. Lett. **1** (1982), 59–62.

Traub, J. F. and Woźniakowski, H. [84], *On the optimal solution of large linear systems*, J. Assoc. Comput. Mach. **31** (1984), 545–559.

Traub, J. F. and Woźniakowski, H. [91a], *Theory and Applications of Information-Based Complexity*, 1990 Lectures in Complex Systems, Santa Fe Institute,, 163–193, Addison-Wesley, Reading, Mass., 1991.

Traub, J. F. and Woźniakowski, H. [91b], *Information-Based Complexity: New Questions for Mathematicians*, Math. Intell. **13** (1991), 34–43.

Wasilkowski, G. W. [89a], *A Clock Synchronization Problem with Random Delays*, J. Complexity **5** (1989), 1–11.

Wasilkowski, G. W. [89b], *Randomization for Continuous Problems*, J. Complexity **5** (1989), 195–218.

Wasilkowski, G. W. [93], *Integration and approximation of multivariate functions: average case complexity with isotropic Wiener measure*, Bull. Amer. Math. Soc. (N.S) **28** (1993), 308–314.

Wasilkowski, G.W. and Woźniakowski, H. [93], *There exists a linear problem with infinite combinatory complexity*, J. Complexity **9** (1993), 326–337.

Wasilkowski, G.W. and Woźniakowski, H. [94], in progress (1994).

Werschulz, A. G. [91], *The Computational Complexity of Differential and Integral Equations*, Oxford University Press, Oxford, 1991.

Woźniakowski, H. [75], *Generalized information and maximal order of iteration for operator equations*, SIAM J. Numer. Anal. **12** (1975), 121–135.

Woźniakowski, H. [91], *Average Case Complexity of Multivariate Integration*, Bull. Amer. Math. Soc. (N.S) **24** (1991), 185–194.

Woźniakowski, H. [92a], *Complexity of Verification and Computation for IBC Problems*, J. Complexity **8** (1992), 93–123.

Woźniakowski, H. [92b], *Average case complexity of linear multivariate problems, Part 1: Theory, Part 2: Applications*, J. Complexity **8** (1992), 337–372, 373–392.

Woźniakowski, H. [93], *Tractability and Strong Tractability of Linear Multivariate Problems*, in progress (1993).