INFORMATION-BASED COMPLEXITY

For articles on related subjects *see* COMPUTATIONAL COMPLEXITY; and NP-COMPLETE PROBLEMS

Many of the mathematical models used in fields such as physical sciences, engineering, economics, and mathematical finance use continuous mathematical models. These models typically require the numerical solution of multivariate problems (often in a very large number of variables) such as integrals, ordinary and partial differential equations, optimization, approximation, integral equations, and nonlinear equations. The study of the computational complexity of continuous mathematical problems is called information-based complexity (IBC). This is a branch of *computational complexity*, which studies the minimal computer resources (typically time or space) needed to solve mathematically posed problems.

Traub and Werschulz (1998) provides an expository introduction to the theory and applications of IBC with a bibliography of over 400 recent papers and books. A general formulation and extensive bibliography can be found in Traub, Wasilkowski, and Woźniakowski (1988).

Computational Complexity of Integration We illustrate some of the important ideas of information-based complexity with the example of univariate integration. (To see how IBC can be used for the numerical computation of the very high dimensional integrals occurring in practice, see the section on **Mathematical Finance** below.) Consider the computation of $\int_0^1 f(x)dx$. For "most" integrands, this integral cannot be computed symbolically. Therefore, we must integrate numerically. All we can enter into the computer are some numbers that represent $f(x)$. Often, the available information is the values of f at a number of points. This information is *partial* because there are many integrands that are indistinguishable using this information. If the information has errors (due, for example, to roundoff errors), it is *contaminated*. It is clear that partial or contaminated information causes *uncertainty*, say ε, in the integral. This uncertainty is intrinsic and caused by the limited information. Furthermore, we assume that the information is priced.

To introduce computational complexity, we first define the *model of computation*. The model of computation states which operations are permitted and how much they cost. The model of computation is based on two assumptions:

1. We can perform arithmetic operations and comparisons on real numbers at unit cost.
2. We can perform an information operation at cost c.

We comment on these assumption. The *real number model* (Assumption 1) is used as an abstraction of the floating-point model typically used in scientific computation. Except for the possible effect of roundoff errors and numerical stability, complexity results will be the same in these two models.

The real number model should be contrasted with the *Turing Machine (q.v.) model*, typically used for discrete problems. The cost of an operation in a Turing Machine model depends on the size of the operands, which is one of the reasons that we use the real number model. For a full discussion of the pros and cons of the Turing Machine and the real number models see Chapter 8, Traub and Werschulz (1998). Whether the real number or Turing Machine model is used can make an enormous difference. For, example, Kachian showed that linear programming is polynomial in the Turing Machine model. In 1981, Traub and Woźniakowski showed that Kachian's algorithm is not polynomial in the real number model and conjectured that linear programming is not polynomial in this model. This conjecture is still open.

In Assumption 2, c is typically much larger than one. The purpose of information operations is to replace the input by a finite set of numbers. For integration, the information operations are typically function evaluations.

A *combinatorial algorithm* is any procedure for approximating the integral using the available information. Any such algorithm must have error at least as large as the intrinsic uncertainty. The *computational complexity* of the integration problem is the minimal cost of computing the integral to within error ε. The *information complexity* and the *combinatorial complexity* of the integration problem are the minimal information and combinatory cost of calculating the integral to within error ε.

The concept of computational complexity permits us to introduce the fundamental concepts of optimal information and optimal algorithm. Information and an algorithm that uses the information are called *optimal information* and *optimal algorithm* if the error of the approximation is at most the intrinsic uncertainty and the cost of computing the approximation equal the computational complexity.

General Formulation We have used the integration example to introduce basic concepts of IBC. Generally, IBC is characterized by three assumptions, namely that information is *partial*, *contaminated* and *priced*. A general formulation and extensive bibliography may be found in Traub, Wasilkowski, and Woźniakowski (1988).

IBC has been used for numerous problems including partial differential equations, path integrals, approximations, nonlinear optimization, and integral equations. There has also been some work on finite-dimensional problems such as large linear systems and eigenvalue problems, as well as the discrete problem of synchronizing clocks in a distributed system (*q.v.*). Fields where IBC has been applied include economics, computer graphics, computer vision (*q.v.*), control theory, and mathematical finance.

Intractable Problems In applications, the inputs are often multivariate functions. For example, we might want to integrate a function of d variables. If the smoothness of the class of inputs is r, then to guarantee an error of at most ε,

$$\mathrm{comp}^{\text{wor-det}}(\varepsilon) = \Theta\left(\varepsilon^{-d/r}\right).$$

The symbol on the left-hand side indicates this is the complexity in the worst case deterministic setting. Worst case means that we guarantee an error of at most ε for every input in a class of inputs. The capital theta notation means that the upper and lower bounds on the computational complexity differ by at most a factor independent of ε.

We comment on this result:

1. It has been proven to hold for many problems, including integration, approximation, nonlinear optimization, systems of nonlinear equations, linear elliptic differential equations, and Fredholm integral equations of the second kind.
2. For ε and r fixed, the complexity is exponential in d.
3. For fixed values of the smoothness r and the error threshold ε, the complexity depends exponentially on the dimension. In 1957, Richard Bellman called exponential dependence on dimension the *curse of dimensionality*. This was based on experience, not theorems of computational complexity. All the problems listed in 1 suffer from this curse in the worst case deterministic setting.
4. In complexity theory we say that a problem with this complexity is *intractable*. We can regard the dimension as the *size* of input. (The number of variables and dimension are used interchangeably.) This may be contrasted with discrete problems, where the size of the input is typically the number of objects.
5. The worst case deterministic setting is the same as the one used in the theory of *NP-complete* problems.
6. In IBC we are often able to obtain tight complexity bounds. This may be contrasted with combinatorial problems such as the traveling salesman problem (see COMPUTATIONAL COMPLEXITY) where only a complexity hierarchy is available and it is usually not known if problems have, say, polynomial or exponential complexity. Since the information in IBC is partial and/or contaminated we are able to obtain lower bounds using adversary arguments at the information level. This is why this field is called *information-based complexity*.
7. However, there are IBC problems where the combinatory complexity dominates the information complexity and we cannot obtain tight complexity bounds. One example is a problem of decentralized control. Papadimitriou and Tsitsiklis have shown that the information complexity is a polynomial in $1/\varepsilon$ but that the combinatorial complexity is a polynomial in $1/\varepsilon$ iff P=NP.
8. Even for rather small values of d, the problem may be very hard. For example, let $\varepsilon = 10^{-8}$ (single-precision), $r = 1$, $d = 3$. Then the complexity for multivariate integration is proportional to 10^{24} function evaluations.

Very high dimensional problems occur in supercomputing and in the foundations of physics. For example, computational chemistry, computational design of pharmaceuticals, and computational metallurgy involve computation with huge numbers of particles. Economic modeling can involve a large number of variables. High dimensional problems occur in mathematical finance. Path integrals, which are of great importance in physics, are infinite-dimensional, and therefore, invite high dimensional approximations.

This motivates our interest in breaking the curse of dimensionality. Since this is a complexity result, we cannot get around it by a clever algorithm. We can try to break the curse by settling for a stochastic assurance rather than a worst case deterministic assurance. Examples of stochastic assurance are provided by the randomized and average case settings which we consider in turn. We can also try to break the curse by changing the class of the inputs; we'll use mathematical finance to illustrate this.

Randomization The first significant use of randomization was the Monte Carlo method (*q.v.*), introduced in the 1940s. As indicated above, multivariate integration is exponential in dimension if coin tossing is not permitted, even if the integrand is evaluated at optimal points. However, if the integrand is sampled at random points,

$$\mathrm{comp}^{\mathrm{wor-ran}}(\varepsilon) = \Theta\left(\varepsilon^{-2}\right)$$

Hence, the symbol on the left-hand side indicates this is the complexity in the worst case randomized setting.

This is a truly remarkable result. If one evaluates at random rather than at optimal deterministic points, the complexity is independent of the number of variables d ! Furthermore, this holds even if the class of integrands consists of functions that are only continuous (i.e., $r = 0$). For this class, $\mathrm{comp}^{\mathrm{wor\text{-}det}}(\varepsilon) = \infty$.

Thus, randomization can be very powerful for continuous problem, just as for discrete problems (see COMPUTATIONAL COMPLEXITY). Not all intractable problems can be broken by using randomization. It is an open question to characterize the class of problems for which intractability is broken by randomization.

Average Case Another way we can try to break intractability is to settle for an average case assurance. That is we guarantee that the *expected* error is at most ε and the computational complexity is the minimal *expected* cost.

Recall that for multivariate integration

$$\mathrm{comp}^{\mathrm{wor\text{-}det}}(\varepsilon) = \Theta\left(\varepsilon^{-d/r}\right)$$

A question that has been open for 20 years, until settled by Woźniakowski in 1991, is the complexity of integration in the average case deterministic setting. Let the class of inputs be continuous functions, equipped with the Wiener measure. (The Wiener measure is a special kind of Gaussian measure, which is a generalization of a Gaussian distribution.) Let the domain of integration be the unit cube in d dimensions. Woźniakowski showed that

$$\mathrm{comp}^{\text{avg-det}}(\varepsilon) = \Theta\left(\varepsilon^{-1}\left[\log \varepsilon^{-1}\right]^{\frac{d-1}{2}}\right).$$

Woźniakowski also proved that *quasi-Monte Carlo* algorithms were optimal on the average. Despite their name, these algorithms are completely deterministic.

It is an open question to characterize the class of problems for which intractability is broken by the average case setting.

Mathematical Finance We use the important area of mathematical finance to illustrate the complexity issues discussed above and to see how problems in this area force us to consider new issues. The valuation of financial instruments often requires the calculation of very high dimensional integrals. Dimensions of 360 and higher are not unusual. Furthermore, since the integrals can be very complicated requiring between 10^5 and 10^6 floating point operations per integrand evaluation, it is important to minimize the number of evaluations.

What does complexity theory tell us about the computation of high dimensional integrals? Clearly we cannot insist on a worst case deterministic guarantee for integrals of such high dimension. During the last twenty years Monte Carlo methods were used. Using Monte Carlo we can compute the integrals to within an expected error of ε with ε^{-2} function evaluations. Can we do better? Woźniakowski showed that we can compute the integral using quasi-Monte Carlo to within an expected error of ε (where the expectation is with respect to the measure on the space of integrands), using just ε^{-1} function evaluations times the polylog factor $\left[\log \varepsilon^{-1}\right]^{\frac{d-1}{2}}$. The famous theorem of Koksma-Hlawka has a similar polylog factor.

For d fixed and ε going to zero, quasi-Monte Carlo converges faster than Monte Carlo. But in mathematical finance ε is typically between 10^{-2} and 10^{-4}, which is not very small, and the dimension d is huge, say, 360 and higher. For such values of d the polylog factor looks ominous. Yet computer experimentation on real-world financial problems, see Paskov and Traub (1995), revealed that quasi-Monte Carlo beats Monte Carlo consistently and by large margins.

The existing theory does not explain the superiority of quasi-Monte Carlo for problems of mathematical finance. What property characterizes the integrands? They are highly non-isotropic; that is, some variables are far more important than others. This property is formalized by Sloan and Woźniakowski (1998), who proved the existence of quasi-Monte Carlo methods whose convergence is independent of the dimension for integrands with this property. Furthermore, this good convergence holds even with a worst case deterministic guarantee. We see that an alternative to changing the setting in order to avoid the curse of dimensionality is to change the set of inputs, in this application, the set of integrands. See Chapter 4 of Traub and Werschulz (1998) for a survey of high dimensional integration and mathematical finance.

Computational Complexity for Continuous Problems with Complete Information
We briefly discuss a complexity theory for continuous combinatorial problems; see Blum, Cucker, Shub, and Smale (1998). They consider continuous problems whose inputs are characterized by a finite number of parameters. Thus, they can assume that information is complete and exact. They adopt the real number model. One of their results is that the problem of determining whether a system of n real polynomials of degree at most four has a real zero is NP-complete.

Fig. 1 schematizes the relation between IBC and other branches of computational complexity.

References

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

1991. Werschulz, A. G. *The Computational Complexity of Differential and Integral Equations: An Information-Based Approach*. New York: Oxford University Press.

1995. Traub, J. F., Paskov, S. "Faster Valuation of Financial Derivatives." *Journal of Portfolio Management* **22** (1), 113-120.

1996. Plaskota, L. *Noisy Information and Computational Complexity*. Cambridge, UK: Cambridge University Press.

1998. Blum, L., Cucker, F., Shub, M., and Smale, S. *Complexity and Real Computation*. New York: Springer-Verlag.

1998. Sloan, I., Woźniakowski , H. "When are Quasi-Monte Carlo Algorithms Efficient for High Dimensional Integrals." *Journal of Complexity* **14** (1), 1-33.

1998. Traub, J. F., Werschulz, A. G. *Complexity and Information*. Cambridge, UK: Cambridge University Press.

JOSEPH F. TRAUB

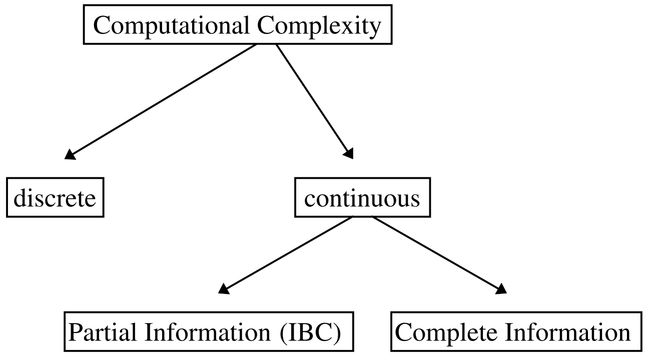

FIG. 1. Structure of computational complexity