Computer Science and Decision Theory: Applications of Notions of Consensus

Description of the Research Issues in the Project

1 Introduction

Many modern computer science problems involve issues that decision theorists have addressed for years, in particular issues involving consensus and associated order relations. Applications of methods of decision theory to problems of computer science place great strain on these methods due to the sheer size of the problems addressed, limitations on information possessed, and sequential nature of repeated applications. Hence, there is great need to develop a new generation of methods to satisfy these requirements of CS applications. In turn, the new methods will provide powerful tools of use in problems of the social sciences (economics, political science, etc.) to which methods of decision theory have traditionally been applied as well as to newer areas of application of decision theory such as in policy-making concerning emerging diseases or bio-terrorism. This project seeks to explore the connections between computer science and decision theory, develop new decision-theory-based methodologies relevant to the scope of modern CS problems, and investigate their applications to problems of computer science and of the social sciences. We shall build the project around notions of consensus that are so central in modern decision theory.

The project is a joint effort between DIMACS, the Center for Discrete Mathematics and Theoretical Computer Science, headquartered at Rutgers University, and LAMSADE, the Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision, based at Université Paris IX - Dauphine.

2 Connections between Computer Science Problems and Decision-theoretic Methods of Consensus

The notion of consensus arises in many decision making applications when we try to combine different opinions to reach one "consensus" opinion. The opinions can be stated in various ways, e.g., as "first choices" among a set of alternatives or as "rankings" (orders) on the alternatives. Here, we describe some of the research issues to be investigated.

2.1 Meta-search, Collaborative Filtering

In CS, "meta-search" is an example of a consensus problem, involving combining the results of several search engines (see [31]). Cohen, Schapire, and Singer [22] showed how meta-search can be formulated as a problem of consensus. They studied the problem of constructing a new ordering based on a learned probabilistic classifier or regression model, showed that the problem of finding the ordering that best agrees with a learned preference function is NP-complete, and described a simple greedy algorithm guaranteed to find a good approximation. Similar ideas of consensus have been used to address the reduction of spam, through the combination of results from multiple search engines. Dwork, et al. [30] think of a ranking function as spammed by a page in a database of pages if in response to a given query it ranks the page too highly. Dwork, et al. showed that the consensus approach involving Kemeny medians ([49, 50, 81, 84]), widely used in social science applications, has excellent "spam-resistant" properties. Unfortunately, it is well known that the computation of the Kemeny median is NP-complete ([8, 101]). Dwork, et al. developed an approximation to the Kemeny median that preserves most of its desirable properties and is computationally tractable. We will investigate approximations to other consensus functions in various computer science contexts. Analogous problems arise in information retrieval, when we try to rank documents according to their probability of relevance to a query.

Consensus methods also arise in collaborative filtering, where we use knowledge about the behavior of multiple users to make recommendations to another user, e.g., combining book or movie ratings to prepare an ordered list of books or movies to recommend ([38, 72, 74, 77]). In these applications, what is different from the traditional decision-theoretic problem is that the number of “voters" is small and the number of “alternatives" or “candidates" is large. Moreover, quite often the problem consists of assigning “alternatives” to pre-defined ordered or nominal categories, a problem which is different from the classical ranking problem studied in decision theory. All the above call for new methods and algorithms.

2.2 Large Databases and Inference

Consensus methods also arise in applications involving large databases, e.g., databases of molecular sequences, when we seek to choose sequences that occur frequently or are “centrally located." (For an example of such work, see [63].) This problem is one example of the emerging field of “bioconsensus" that involves applications of social-science-based consensus methods to problems of biology, many arising from the modern information-theoretic analysis of bioinformatics. Day [23] has compiled a bibliography of many papers in this emerging area and “bioconsensus" has been the subject of several DIMACS research working groups (see http://dimacs.rutgers.edu/Workshops/Bioconsensus/ and http://dimacs.rutgers.edu/Workshops/BioconII/ and the books [24, 46]. We will explore various methods and models in bioconsensus, including consensus ideas motivated by the notion of list coloring in graph theory [58].

Similar problems arise in “homeland security" applications. For instance, in the DIMACS project on “Monitoring Message Streams" (http://www.stat.rutgers.edu/ madigan/mms/), one deals with large sets of text messages and attempts to classify them by content. Methods of “fusion" of the results of several different classification algorithms are playing an important “consensus" role in that project (see [4]. (See also [44, 66, 51, 10, 9].) The problem has been studied within Artificial Intelligence approaches where decision theory has already introduced several major contributions (see [54], [52], [21], [5], [53, 80, 19, 29]). Voting methods can help to achieve high accuracy in learning algorithms by combining the predictions of several classifiers ([73, 91]). Decision theoretic approaches have also been used in order to induce classification rules from (partially) inconsistent and or incomplete data bases ([68]). For instance the LAMSADE project “Mathematical and Logical Structures for Data Mining, Preference Modelling and Decision Aiding” (see http://www.lamsade.dauphine.fr/mcda/siteTheme/p1.html) is dedicated to this problem ([94, 93, 95, 96]). How can we take advantage of new procedures for combination to build better learning algorithms? We will investigate this type of question.

2.3 Software and Hardware Measurement

Combining different measures or ratings through appropriate consensus methods is an important topic in the measurement of the understandability, quality, functionality, reliability, efficiency, usability, maintainability, or portability of software [13, 32, 60]. Methods of measurement theory ([82, 83, 88]) and of multiple criteria decision theory [100] have been used here with interesting results as shown within a LAMSADE dedicated project (see http://www.lamsade.dauphine.fr/mcda/siteTheme/p8.html, [67, 64, 65, 12]). However, new methods are needed to deal with complexities arising from CS-based applications. Similar measurement issues arise in hardware measurement [35] and collaborative filtering [72] and work on combining “normalized scores" [1, 85] is relevant. We will investigate relevant decision-theoretic methods of measurement theory and multiple criteria decision theory.

2.4 Consensus Computing, Image Processing

An important subject in the social sciences involves models of how opinions change over time, until hopefully some consensus is reached. Models include neural nets, threshold models, and Markov chains [25, 36, 81, 41, 79, 27]. These models have begun to be applied in CS applications in distributed computing [43, 55, 70, 71]. In such applications, the values of processors in a network are updated until all the processors have the same value.

One application of this idea is in noise removal in digital images [45]. To check if a pixel is noise, one compares it with neighboring pixels. If the values are beyond a certain threshold, one replaces the value of the given pixel with a mean or median of the values of its neighbors.

Related models arise in “distributed consensus" [69] where non-faulty processors are required to reach agreement eventually. Good protocols for how this can be arranged, which include accepting values (“opinions") of neighboring non-faulty processors through some sort of majority rule process, need to be developed. In [11], a protocol based on the parliamentary procedure known as “cloture” is shown to be very good in terms of a number of important criteria including polynomial computation and communication. Much of the work on opinion-change models has centered around the question of identifying initial configurations of opinions or opinion reformulation rules that lead to ultimate consensus. We will seek to develop good procedures for answering this question in models arising in CS contexts.

Similar methods show promise for modern issues in epidemiology, where we deal with large models involving social networks and the spread of disease (and “opinion" is replaced by having or not having the disease). Here, a key problem is to identify initial configurations of vaccinated individuals so that no matter how many individuals are initially infected with a disease, the ultimate “consensus" will have no one with the disease. Some of the work in this area is motivated by newly emerging diseases and some by diseases deliberately spread by bioterrorists. For some work along these lines, see [28]; see [90] for discussion of the role of such methods in defense against bioterrorism. We will pursue these epidemiological applications.

2.5 Computational Intractability of Consensus Functions

Many consensus functions are known to be computationally intractable [6, 7, 8]. Especially in the CS context, work is needed to develop good algorithms, efficient approximations, and tractable heuristic methods. (See [30, 59] for some work along these lines.) Sometimes computational intractability is a “good thing," for example when we try to design voting systems where it is computationally intractable to manipulate the outcome of an election by “insincere" voting, adding voters, declaring voters ineligible, adding candidates, declaring candidates ineligible, or adjusting the order in which alternatives are considered in a multiple-step voting situation. (Early work on these issues is contained in [7].) New methods we will investigate in this project could help with such problems. In turn, new methods developed in these CS applications should be useful in making it difficult to manipulate the outcomes of electronic voting systems.

2.6 Axiomatic Approaches and Algorithms

The size of modern CS applications requires new algorithms, approximations, etc. since existing algorithms don’t scale well. An intriguing idea is that the axiomatic approach, a standard tool in consensus theory, could help here. The functional form of a consensus function can sometimes constrain the choice of algorithms. A large part of the research done at LAMSADE is dedicated to such problems ([98, 14, 15, 16, 18, 17], see also [99]). There has also been extensive work at DIMACS in this area ([34, 42, 61, 62, 86, 87]). Axiomatic characterizations of consensus functions can help us in identifying these functional forms without necessarily specifying the exact consensus function, and hence in turn help us to develop algorithms for calculating consensus. Totally new functional forms for consensus functions, derived with computer science applications in mind, can sometimes result from an axiomatic approach (see, e.g., [72] in an IT framework and [2, 3] in more general contexts). We shall pursue this approach.

2.7 Order Relations and Revealed Preferences

Establishing a consensus implies being able to compare objects either in order to establish that one is “before" the other or that they are “near." Concepts developed or used in decision theory (non-symmetric similarities, special kinds of partial orders such as interval orders or semiorders, etc. [33, 78, 97]) should be useful here and have already found application in computer science. Applications of ordered sets in computer science (see for instance [92]) include their uses as models for computation, their applications in knowledge representation, text categorization and data mining, and their uses in analyzing crypto-protocols in security and in inductive logic programming. (For examples of such applications, see the DIMACS workshop on Applications of Lattices and Ordered Sets to Computer Science (http://dimacs.rutgers.edu/Workshops/Lattices/). Issues of special interest in this project include the way in which preferences are revealed before a consensus is reached (see also the special LAMSADE project on such issues: http://www.lamsade.dauphine.fr/mcda/siteTheme/p4.html). In traditional models of consensus, such preferences are thought of as various kinds of orders that are all revealed at the beginning (perfect information). But increasingly in CS applications, especially in situations of economic cooperation and competition using the Internet, we can think of software agents learning about other players' preferences through repeated or iterated auctions or other procedures. By learning about a competitor's preferences and, ultimately, its utility function, without seriously intending to win an auction, a firm can then seriously enter a later auction with a significant information advantage. Work of [47, 37, 40, 39, 48] is relevant. Building models for consensus in this new context is another challenge we shall pursue. A related topic building on order relations concerns the issue of solving classic optimization problems when the costs are expressed in ordinal scales (not allowing an additive problem formulation), a problem receiving increasing attention in the planning and scheduling community of CS (see [26], [56], [57], [76], [75], [89]).