The most dramatic interaction between CS and GT may involve game-theory pragmatics.

BY YOAV SHOHAM

Computer Science and Game Theory

GAME THEORY HAS influenced many fields, including economics (its initial focus), political science, biology, and many others. In recent years, its presence in computer science has become impossible to ignore. GT is an integral part of artificial intelligence (AI), theory, e-commerce, networking, and other areas of computer science, and it is routinely featured in the field's leading journals and conferences. One reason is application pull: the Internet calls for analysis and design of systems that span multiple entities, each with its own information and interests. Game theory, for all its limitations, is by far the most developed theory of such interactions. Another reason is technology push: the mathematics and scientific mind-set of game theory are similar to those that characterize many computer scientists. Indeed, it is interesting to note that modern computer science and modern game theory originated in large measure at the same place and time—namely at Princeton

University, under the leadership of John von Neumann, in the 1950s.[a]

In this article I try to do two things: identify the main areas of interaction between computer science and game theory so far; and point to where the most interesting interaction yet may lie—in an area that is still relatively underexplored.

The first part aims to be an unbiased survey, but it is impossible to avoid bias altogether. Ten researchers surveying the interactions between CS and GT would probably write 10 different types of reports. Indeed, several already have (as I will discuss). Moreover, in this brief discussion I cannot possibly do justice to all the work taking place in the area. So I try to compensate for these limitations in two ways: I provide a balanced set of initial pointers into the different subareas, without regard to the amount or nature of work that has taken place in each; and I point the reader to other relevant surveys of the CS-GT interaction, each having its own take on things.

The second part is decidedly subjective, but it is still meant to be broadly relevant both to computer scientists and game theorists interested in the interaction between the disciplines.

Lessons from Kalai (1995)

My departure point is a 13-year-old survey paper by E. Kalai,[16] a game theorist with algorithmic sensibilities. Geared primarily toward computer scientists, the paper took stock of the interactions between game theory, operations research, and computer science at the time. It points to the following areas:

1. Graphs in games
2. The complexity of solving a game
3. Multiperson operations research
4. The complexity of playing a game
5. Modeling bounded rationality.

The reason I start with this paper, besides providing the interesting perspective of a non-computer scientist, is the comparison with current CS-GT interac-

---

a   I thank Moshe Tennenholtz for this observation, which is especially true of GT and AI.

tion, as both the matches and mismatches are instructive. Looking at the interactions between CS and GT taking place today, one can identify the following foci:

- a. Compact game representations;
- b. Complexity of, and algorithms for, computing solution concepts;
- c. Algorithmic aspects of mechanism design;
- d. Game-theoretic analysis inspired by specific applications;
- e. Multiagent learning;
- f. Logics of knowledge and belief, and other logical aspects of games.[b]

The crude mapping between this list and Kalai's is as follows:

1995		2008
1	◄ ►	a
2	◄ ►	b
3	◄ ►	c, d
4		
5		
		e,
		f

Here, I discuss the areas that match up (1◄ ►a, 2◄ ►b, 3◄ ►c, d), then turn to the currently active areas that were not discussed by Kalai (e, f), and finish with the orphans on the other side (4, 5) that were discussed by Kalai but not yet vigorously pursued.

There has been substantial work on compact and otherwise specialized game representations. Some of them are indeed graph-based—graphical games,[18] local-effect games,[21] MAIDS,[19] and Game networks,[20] for example. The graph-based representations extend also to coalition game theory.[7] But specialized representations exist that are not graph based, such as those that are multi-attribute based[5] and logic based.[15] I believe this area is ripe for additional work—regarding, for example, the strategy space of agents described using constructs of programming languages.

The complexity of computing a sample Nash equilibrium (as well as other solution concepts) has been the focus of much interest in CS, especially within the theory community. A new complexity class—PPAD—was proposed to handle problems for which a solution is known to exist,[27] the computation of a sample Nash equilibrium was shown to be complete for this class,[2] and the problem of computing Nash equilibria with specific properties was shown to be NP-hard.[4, 10] At the same time, algorithms—some quite sophisticated, and all exponential in the worst case—have been proposed to compute Nash equilibria.[11, 41] Somewhat surprisingly, recent experiments have shown that a relatively simple search algorithm significantly outperforms more sophisticated algorithms.[31] This is an active area that promises many additional results.

The third match is somewhat less tight than the first two. There are at least two kinds of optimization one could speak about in a game-theoretic setting. The first is computing a best response to a fixed decision by the other agents; this is of course the quintessential single-agent optimization problem of operations research and AI, among other fields. The second is the optimization by the designer of a mechanism aimed at inducing games with desirable equilibria.

This so-called "mechanism design" has been the focus of much work in computer science. One reason is the interesting interaction between traditional CS problems (such as optimization and approximation) and traditional mechanism-design issues (such as incentive compatibility, individual rationality, and social-welfare maximization). A good example is the interaction between the Vickrey-Clarke-Groves mechanism and shortest-path computation;[26] another is the literature on combinatorial auctions,[6] which combine a weighted-set-packing-like NP-hard optimization problem with incentive issues. The interplay between mechanism design and cryptography is worth particular mention. Though both are in the business of controlled dissemination of information, they are different in significant ways. For one thing, they are dual in the following sense: mechanism design attempts to *force* the *revelation* of information, while cryptography attempts to *allow* its *hiding*. For another, they traditionally embody quite different models of paranoia. Game theory assumes an even-keeled expected utility maximization on the part of all agents, while cryptography is more simple-minded: it assumes that "good" agents act as instructed, while "bad" agents are maximally harmful. Recent work, however, has begun to bridge these gaps.

This third category blends into the fourth one, which is research motivated by specific applications that have emerged in the past decade. For example, the domain of networking has given rise to a literature on so-called "price of anarchy" (which captures the inefficiency of equilibria in that domain), games of routing, networking-formation games, and peer-to-peer networks. Other domains include sponsored search auctions, information markets, and reputation systems. This combination of the third and fourth categories is arguably the most active area today at the interface of CS and GT, and many aspects of it are covered in Nisan et al.,[25] which is an extensive edited collection of surveys. The popularity of this area is perhaps not surprising. The relevancies of specific applications speak for themselves (although arguments remain about whether the traditional game-theoretic analysis is an appropriate one). More generally, it is not surprising that mechanism design struck a chord in CS, given that much of CS's focus is on the design of algorithms and protocols. Mechanism design is the one area within GT that adopts such a design stance.

The fifth category active today is multiagent learning, also called "interactive learning" in the game-theory literature.[c] Multiagent learning, long a major focus within game theory, has been rediscovered with something of a vengeance in computer science and in particular AI; witness special issues devoted to it in the *Journal of Artificial Intelligence*[39] and the *Machine Learning Journal*.[12] For computer science, the move from single-agent learning to multiagent learning is interesting not only because it calls for new solutions but also because the very questions change. When multiple agents learn concurrently, one cannot distinguish between learning and teaching, and the question of "optimal" learning is no longer well defined (just as the more general notion of an "optimal policy" ceases to be meaningful when one moves to the multiagent setting). For a discussion of this phenomenon, see the *Journal of Artificial Intelligence* special issue cited earlier.[39]

---

b This current survey originated in a presentation made at a December 2007 festschrift in honor of E. Kalai.

c Kalai's omission of this area is ironic, as he co-authored one of its seminal papers.

The sixth and final major area of focus, also one not discussed in Kalai,[16] is called "interactive epistemology" in game theory and simply "reasoning about knowledge and belief" in computer science. Starting in the mid-1980s, this area was for a while the most active focus of interaction between computer science (including distributed systems, AI, and theory) and game theory. Beside game theory, it established deep ties with philosophy and mathematical logic, culminating in the seminal book by Fagin et al.[8, d] It is interesting to speculate why this area was omitted from Kalai's list, even although it predates his paper by a decade, and why today it is not as broadly populated as the other areas. I think the reason is that the subject matter is more foundational, primarily non-algorithmic, and appeals to a smaller sliver of the two communities. Be that as it may, it remains a key area of interaction between the two fields.

These six areas are where most of the action has been in past years, but by listing only them and being brief about each one, I have by necessity glossed over some other important areas. The references compensate for this omission to some extent. In addition, the reader is referred to the following additional surveys, all by computer scientists who each have a slightly different slant. Most of these works go into considerably more detail about some of the topics.

▸ The earliest relevant survey is probably by Linial.[22] Geared primarily toward game theorists, this 58-page report has deep coverage of game-theoretic aspects of distributed systems, fault-tolerant computing, and cryptography, and it also touches on computation of game theoretic concepts, games and logic, and other topics.

▸ Papadimitriou's survey[29] geared to-

d That book focused on static aspects of knowledge and belief, which, notwithstanding the substantial computer-science credentials of the authors, raise an interesting contrast between the computer-science and game-theory literature in these areas. In game theory, static theories are indeed the primary focus, whereas in computer science—in particular, in database theory and artificial intelligence—belief revision and other dynamic theories[30] (including the entire mini-industry of nonmonotonic logics[9]) play an equal if not greater role. Indeed, recent work at the interface of logic and game theory[37] extends the static treatment of Fagin et al.[8] in a dynamic direction.

I expect game-theory pragmatics to be as critical to reducing game theory to practice as language pragmatics have been to analyzing human discourse or understanding language by computers.

ward computer scientists, is a concise five-page paper summarizing the main complexity and algorithmic issues at the interface of CS and GT circa 2001.

▸ The 21-page paper by Halpern[13] is similar to Linial in that it is geared toward game theorists and its main focus is distributed systems, but having been published a decade later it is more current. The work later evolved into a 17-page survey[14] with an abbreviated discussion of distributed computing and additional material on complexity considerations, price of anarchy, mediators, and other topics.

▸ Roughgarden's 30-page work is a detailed survey of a specific topic—namely, the complexity of computing a sample Nash equilibrium.[32] Geared mostly toward economists, it includes ample background material on relevant concepts from complexity theory.

The material discussed so far is not only prominently featured in computer science journals and conferences but also is beginning to find its way into textbooks.[35] These areas will undoubtedly continue to flourish. But now I want to turn our attention to the two closely-related areas—4 and 5—listed by Kalai that have *not* been looked at as closely by the community at large, CS in particular. I do this for two reasons: I believe they are critical to the future success of game theory, and I believe that CS can play an important role in them. They both have to do with incorporating practical considerations into the model of rationality that is inherent to game theory. To repeat the caveat stated earlier: unlike the material so far, the remaining discussion is future-directed, speculative, and subjective.

Lessons from Linguistics

The field of linguistics distinguishes among syntax, semantics, and pragmatics. Syntax defines the form of language, semantics defines its meaning, and pragmatics defines its use. While the three interact in important ways, the distinctions have proved very useful. I believe that game theory may do well to make similar distinctions, and that CS can help in the process. Just as in the case in linguistics, it is unlikely that game-theory pragmatics will yield to unified clean theories, as do syntax and semantics. But I expect game-theory pragmatics to be as critical to reduc-

ing game theory to practice as language pragmatics have been to analyzing human discourse or understanding language by computers.

The distinction between the syntax and semantics of games is, I think, quite important, as some of the disputes within game theory regarding the primacy of different game representations (for example, the strategic and extensive forms) suffer from the lack of this distinction. It might, however, be presumptuous for CS to intrude on this debate, except insofar as it lends logical insights.[38] Indeed, perhaps this is more the role of mathematical logic than of CS per se.

But where CS can truly lead the way is in game theory's pragmatics. Game theory as we know it embodies radical idealizations, which include the infinite capacity of agents to reason and the infinite mutually recursive modeling of agents. Backing off from these strong assumptions has proven challenging. A fairly thin strand of work under the heading of "bounded rationality" involves games played by automata.[33] This is an important area of research that sometimes makes deep connections between the two fields. For example, early results showed that one of the well-known pesky facts in game theory—namely, that constant "defection" is the only subgame-perfect equilibrium in the finitely repeated prisoner's dilemma game—ceases to hold true if the players are finite automata with sufficiently few states.[24, 28] A more recent result shows that when players in a game are computer programs, one obtains phenomena akin to the Folk Theorem for repeated games.[36]

This connection between theoretical models of computation and game theory is quite important and beautiful, but it constitutes a fairly narrow interpretation of the term "bounded rationality." The term should perhaps be reserved for describing a much broader research agenda—one that may encourage more radical departures from the traditional view in game theory. Let me mention two directions that I think would be profitable (and difficult) to pursue under this broader umbrella.

When one takes seriously the notion of agents' limited reasoning powers, it is not only some of the answers that begin to change; the questions themselves must be reconsidered. Consider

> **Science operates at many levels. For some, it is sufficient that scientific theories be clever, beautiful, and inspirational. Others require that any science eventually make contact with compelling applications and be subjected to empirical evaluation.**

the basic workhorses of game theory—the Nash equilibrium and its many variants—that have so far served as the very basic analysis tool of strategic interactions. Questioning the role of equilibrium analysis will be viewed by some in GT as act of heresy, but real life suggests that we may have no choice. For example, in the trading agent competition, Nash equilibrium of the game did not play a role in almost any participating program,[42] and this is certainly true as well of the more established chess and checkers competitions.

It is premature to write off the Nash equilibrium as irrelevant, however. For example, two programs competing in the TAC did in fact make use of what can be viewed as approximate empirical NE.[42] Another striking example is the computation of equilibria in a simplified game tree by a top-scoring program in a poker competition.[43] It could be argued that maxmin strategies, which coincide with equilibrium strategies in zero-sum games, do play an important pragmatic role. But computation of either maxmin or equilibrium strategies in competitions has certainly been the exception to the rule. The more common experience is that one expends the vast majority of the effort on traditional AI problems such as designing a good heuristic function, searching, and planning. Only a little—albeit important—time is spent reasoning about the opponent.

The impact of such pragmatic considerations on game theory can be dramatic. Rather than start from very strong idealizing assumptions and awkwardly try to back off from them, it may prove more useful or accurate to start from assumptions of rather limited reasoning and mutual modeling, and then judiciously add what is appropriate for the situation being modeled. Which incremental-modeling approach will out has yet to be seen, but the payoff both for CS and GT can be substantial.

The second direction is radical in a different way. Game theory adopts a fairly terse vocabulary, inheriting it from decision theory and the foundations of statistics.[e] In particular, agents

---

e Parenthetically, it can be remarked that Savage's setting,[34] on which the modern Bayesian framework is based, does not have an obvious extension to the multi-agent case. However, this is not the focus of the point I am making here.

have "strategies," which have minimal structure, and motivations, which are encapsulated in a simple real-valued utility function. (This in fact carries even less information than is suggested by the use of numbers, as the theory is unchanged by any positive affine transformation of the numbers.) In real life, and in computer programs attempting to behave intelligently, we find use for a much broader vocabulary. For example, agents are *able* to take certain actions and not others; have *desires*, *goals*, and *intentions* (the belief-desire-intention combination giving rise to the pun "beady-eye agent architecture"); and make *plans*. Apparently these abstract notions are useful both in effecting intelligent behavior and in reasoning about it. Philosophers have written about them (for example, Bratman[1]) and there have been attempts—albeit preliminary ones—to formalize these intuitions (starting with Cohen and Levesque[3]). Some in AI have advocated embracing an even broader vocabulary of emotions (such as the recent provocative albeit informal book by Minsky.[23]) Is game theory missing out by not considering these concepts?

Concluding Remarks
Science operates at many levels. For some, it is sufficient that scientific theories be clever, beautiful, and inspirational. Others require that any science eventually make contact with compelling applications and be subjected to empirical evaluation. Without personally weighing in on this emotional debate, I note that in his 2004 presidential address at the Second World Congress of the Game Theory Society,[17] Kalai reprised the three stages of any science as discussed by von Neumann and Morgenstern:

[W]hat is important is the gradual development of a theory, based on a careful analysis of the ordinary everyday interpretation of economic facts. The theory finally developed must be mathematically rigorous and conceptually general. Its first applications are necessarily to elementary problems where the result has never been in doubt and no theory is actually required. At this early stage the application serves to corroborate the theory. The next stage develops when the theory is applied to somewhat more complicated situations in which

it may already lead to a certain extent beyond the obvious and the familiar. Here theory and application corroborate each other mutually. Beyond this lies the field of real success: genuine predictions by theory. It is well known that all mathematized sciences have gone through these successive phases of evolution.[40]

So at least von Neumann, the father of modern-day game theory and computer science, attached importance to spanning the spectrum from the theoretical to the applied. Pragmatics may be critical to achieving von Neumann and Morgenstern's third stage, and it could propel a joint endeavor between computer science and game theory. ▣

References
1. Bratman, M.E. *Intention, Plans, and Practical Reason.* CSLI Publications, Stanford University, 1987.
2. Chen, X. and Deng, X. Settling the complexity of 2-player Nash–equilibrium. *FOCS*, 2006.
3. Cohen, P.R. and Levesque, H.J. Intention is choice with commitment. *Artificial Intelligence 42*, 2–3 (1990), 213–261.
4. Conitzer, V. and Sandholm, T. Complexity results about Nash equilibria. *IJCAI*. 2003, 761–771.
5. Conitzer, V. and Sandholm, T. Computing Shapley values, manipulating value division schemes, and checking core membership in multi-issue domains. *AAAI*, 2004.
6. Cramton, P.C., Shoham, Y. and Steinberg, R. (eds.) *Combinatorial Auctions*. MIT Press, 2006.
7. Deng, X and Papadimitriou, C.H. On the complexity of cooperative solution concepts. *Mathematics of Operations Research 19*, 257, 1994.
8. Fagin, R., Halpern, J.Y., Moses, Y. and Vardi, M.Y. *Reasoning about Knowledge.* MIT Press, Cambridge, MA, 1995.
9. Niemela, I., Brewka, G., and Truszczynski, M. Nonmonotonic reasoning. In *Handbook of Knowledge Representation*. F. van Harmelen, V. Lifschitz, and B. Porter. (eds.), Elsevier, 2007.
10. Gilboa, I. and Zemel, E. Nash and correlated equilibria: Some complexity considerations. *Games and Economic Behavior*, 80–93, 1989.
11. Govindan, S. and Wilson, R. A global Newton method to compute Nash equilibria. *Journal of Economic Theory*, 2003.
12. Greenwald, A. and Littman, M.L. (eds.). Special issue on learning and computational game theory. *Machine Learning 67*, 1–2, 2007.
13. Halpern, J.Y. A computer scientist looks at game theory. *Games and Economic Behavior 45*, 1 (2003), 114–132.
14. Halpern, J.Y. Computer science and game theory: A brief survey. In *The New Palgrave Dictionary of Economics*. S.N. Durlauf and L.E. Blume (eds.), Palgrave MacMillan, 2008.
15. Ieong, S. and Shoham, Y. Marginal Contribution Nets: A compact representation scheme for coalitional games. In *Proceedings of the 6th ACM Conference on Electronic Commerce*, 2005.
16. Kalai, E. Games, computers, and O.R. In *ACM/SIAM Symposium on Discrete Algorithms*, 1995.
17. Kalai, E. Presidential address. The Second World Congress of the Game Theory Society, Marseille, 2004. *Games and Economic Behavior*, P. Reny (ed.), in press.
18. Kearns, M.J., Littman, M.L., and Singh, S.P. Graphical models for game theory. *UAI*, 2001.
19. Koller, D. and Milch, B. Multiagent influence diagrams for representing and solving games. *IJCAI*, 2001.
20. LaMura, P. Game networks. *UAI*, 2000.
21. Leyton–Brown, K. and Tennenholtz, M. Local–effect games. *IJCAI*, 2003.
22. Linial, N. Game theoretic aspects of computing. In *Handbook of Game Theory*, R.J. Aumann and S. Hart (eds.), 1339–1395, Elsevier Science, 1994.
23. Minsky, M. *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind*, New York: Simon and Shuster, 2007.
24. Neyman, A. Bounded complexity justifies cooperation in finitely repeated prisoner's dilemma. *Economic Letters*, 1985, 227–229.
25. Nisan, N., Roughgarden, T., Tardos, E., and Vazirani, V. (eds.) *Algorithmic Game Theory*. Cambridge University Press, 2007.
26. Nisan, N. and Ronen, A. Algorithmic mechanism design. In *Proceedings of the 31st ACM Symposium on Theory of Computing*, 1999, 129–140.
27. Papadimitriou, C.H. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences 48*, 3 (1004), 498–532.
28. Papadimitriou, C.H. and Yannakakis, M. On bounded rationality and computational complexity. In *Proceedings of the Symposium on the Theory of Computing*, 1994, 726–733.
29. Papadimitriou, C.H. Algorithms, games, and the Internet. *Lecture Notes in Computer Science 2076*, 2001.
30. Pappas, P. Belief revision. In *Handbook of Knowledge Representation*. F. van Harmelen, V. Lifschitz, and B. Porter (eds.). Elsevier, 2007.
31. Porter, R., Nudelman, E., and Shoham, Y. Simple search methods for finding a Nash equilibrium. In *Proceedings of the National Conference on Artificial Intelligence*, 2004, 664–669.
32. Roughgarden, T. Computing equilibria: A computational complexity perspective. *Economic Theory*, 2008.
33. Rubinstein, A. *Modeling Bounded Rationality*. MIT Press, Cambridge, MA, 1998.
34. Savage, L.J. *The Foundations of Statistics*. John Wiley and Sons, NY, 1954. (Second Edition: Dover Press, 1972).
35. Shoham, Y. and Leyton–Brown, K. *Multiagent Systems: Algorithmic, Game Theoretic, and Logical Foundations.* Cambridge University Press, 2008.
36. Tennenholtz, M. Program equilibrium. *Games and Economic Behavior 49*, 2004, 363–373.
37. van Benthem, J. *Exploring Logical Dynamics.* Center for the Study of Language and Information, Stanford, CA, 1997.
38. van Benthem, J. When are two games the same? In *LOFT–III*, 1998 (ILLC preprint, 1999).
39. Vohra, R. and Wellman, M.P. (eds.). Special issue on foundations of multiagent learning. *Artificial Intelligence 171*, 1 (2007).
40. von Neumann, J. and Morgenstern, O. *Theory of Games and Economic Behavior*, Second Edition. Princeton University Press, 1947.
41. von Stengel, B. Computing equilibria for two–person games. In *Handbook of Game Theory*, vol. III, chap. 45. R. Aumann and S. Hart (eds.). Elsevier, Amsterdam, 2002, 1723–1759.
42. Wellman, M.P., Greenwald, A., and Stone, P. *Autonomous Bidding Agents: Strategies and Lessons from the Trading Agent Competition.* MIT Press, 2007.
43. Zinkevich, M., Bowling, M., and Burch. N. A new algorithm for generating equilibria in massive zero–sum games. In *Proceedings of the 22nd Conference on AI*, 2007, 788–793.

Yoav Shoham (http://cs.stanford.edu/~shoham) is a professor of computer science at Stanford University, Stanford, CA.