Cooperating with Machines

Jacob W. Crandall*[1], Mayada Oudah[2], Tennom[2], Fatimah Ishowo-Oloko[2], Sherief Abdallah[3,4],
Jean-François Bonnefon[5], Manuel Cebrian[6], Azim Shariff[7], Michael A. Goodrich[1], and Iyad Rahwan*[8]

[1] *Computer Science Department, Brigham Young University, Provo, UT 84602, USA*
[2] *Masdar Institute of Science and Technology, Abu Dhabi, UAE*
[3] *British University in Dubai, Dubai, UAE*
[4] *School of Informatics, University of Edinburgh, Edinburgh EH8 9AB, UK*
[5] *Toulouse School of Economics, Center for Research in Management, Centre National de la Recherche Scientifique, Institute for Advanced Study in Toulouse, University of Toulouse Capitole, Toulouse, France*
[6] *Data61, Commonwealth Scientific and Industrial Research Organization, Clayton, Victoria 3168, Australia*
[7] *Department of Psychology and Social Behavior, University of California, Irvine, CA 92697, USA*
[8] *The Media Lab, Massachusetts Institute of Technology, Cambridge, MA 02139, USA*

March 22, 2017

Abstract

Since Alan Turing envisioned Artificial Intelligence (AI) [1], a major driving force behind technical progress has been competition with human cognition. Historical milestones have been frequently associated with computers matching or outperforming humans in difficult cognitive tasks (e.g. face recognition [2], personality classification [3], driving cars [4], or playing video games [5]), or defeating humans in strategic zero-sum encounters (e.g. Chess [6], Checkers [7], Jeopardy! [8], Poker [9], or Go [10]). In contrast, less attention has been given to developing autonomous machines that establish mutually cooperative relationships with people who may not share the machine's preferences. A main challenge has been that human cooperation does not require sheer computational power, but rather relies on intuition [11], cultural norms [12], emotions and signals [13, 14, 15, 16], and pre-evolved dispositions toward cooperation [17], common-sense mechanisms that are difficult to encode in machines for arbitrary contexts. Here, we combine a state-of-the-art machine-learning algorithm with novel mechanisms for generating and acting on signals to produce a new learning algorithm that cooperates with people and other machines at levels that rival human cooperation in a variety of two-player repeated stochastic games. This is the first general-purpose algorithm that is capable, given a description of a previously unseen game environment, of learning to cooperate with people within short timescales in scenarios previously unanticipated by algorithm designers. This is achieved without complex opponent modeling or higher-order theories of mind, thus showing that flexible, fast, and general human-machine cooperation is computationally achievable using a non-trivial, but ultimately simple, set of algorithmic mechanisms.

*Correspondence should be addressed to crandall@cs.byu.edu and irahwan@mit.edu

1 Introduction

The emergence of driverless cars, autonomous trading algorithms, and autonomous drone technologies highlight a larger trend in which artificial intelligence (AI) is enabling machines to autonomously carry out complex tasks on behalf of their human stakeholders. To effectively represent their stakeholders in many tasks, these autonomous machines must repeatedly interact with other people and machines that do not fully share the same goals and preferences. While the majority of AI milestones have focused on developing human-level wherewithal to compete with people [6, 7, 8, 9, 10], most scenarios in which AI must interact with people and other machines are not zero-sum interactions. As such, AI must also have the ability to cooperate, even in the midst of conflicting interests and threats of being exploited. Our goal is to understand how to build AI algorithms that cooperate with people and other machines at levels that rival human cooperation in arbitrary two-player repeated interactions.

Algorithms capable of forming cooperative relationships with people and other machines in arbitrary scenarios are not easy to come by. A successful algorithm must satisfy several conditions. First, it must not be domain-specific – it must have superior performance in a wide variety of scenarios (*generality*). Second, the algorithm must learn to establish effective relationships with people and machines without prior knowledge of associates' behaviors (*flexibility*). To do this, it must be able to deter potentially exploitative behavior from its partner and, when beneficial, determine how to elicit cooperation from a (potentially distrustful) partner who might be disinclined to cooperate. Third, when associating with people, the algorithm must learn effective behavior within very short timescales – i.e., within only a few rounds of interaction (*learning speed*). These requirements create many technical challenges (see SI.A.2), the sum of which often causes AI algorithms to fail to cooperate, even when doing so would be beneficial in the long run.

In addition to these computational challenges, human-AI cooperation is difficult due to differences in the way that humans and machines reason. While AI relies on computationally intensive search and random exploration to generate strategic behavior, human cooperation appears to rely on intuition [11], cultural norms [12], emotions and signals [13, 14], and pre-evolved dispositions toward cooperation [17]. In particular, cheap talk (i.e., costless signals) is important to human cooperation in repeated interactions [15, 16], as it helps people coordinate quickly on desirable equilibrium and create shared representations [18, 19, 20, 21]. As such, in addition to generating strategic behavior, we consider that AI algorithms must generate and respond to costless signals at levels that are conducive to human understanding.

2 Results

The primary contribution of this work is the development and analysis of a new learning system that couples a state-of-the-art machine-learning algorithm with novel mechanisms for generating and responding to signals. Via extensive simulations and user studies, we show that this learning system learns to establish and maintain effective relationships with people and other machines in a wide-variety of repeated interactions at levels that rival human cooperation. In so doing, we also investigate the algorithmic mechanisms that are responsible for its success.

2.1 Cooperating with People and Other Machines

Over the last several decades, algorithms for generating strategic behavior in repeated games have been developed in many disciplines, including economics, evolutionary biology, and the AI and machine-learning communities. To begin to evaluate the ability of these algorithms to forge successful cooperative relationships, we selected and evaluated 25 representative algorithms from these fields, including classical algorithms such as (generalized) generous tit-for-tat (i.e., GODFATHER) and win-stay-lose-shift (WSLS) [39], evolutionarily evolved memory-one and memory-two stochastic strategies [40], machine-learning algorithms (including reinforcement learning), belief-based algorithms [41], and expert algorithms [42, 43]. Via extensive simulations, we compared these algorithms with respect to six different performance metrics (see SI.B.2) across the periodic table of 2x2 games [37] (see Methods and SI.A.3).

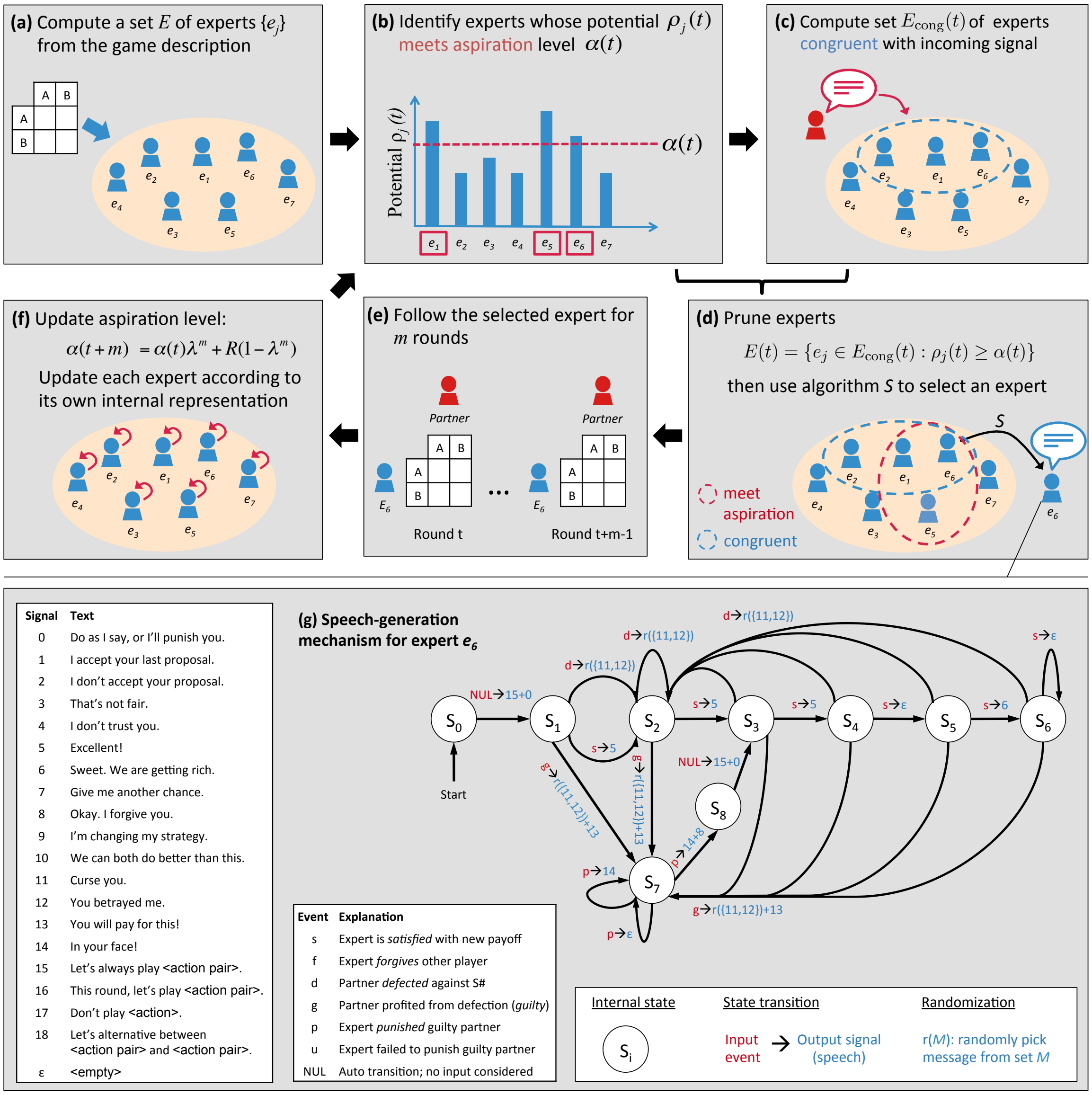
The results of this evaluation, which are overviewed in Methods (see Figure 6 in particular) and are described in detailed in SI.B, demonstrate the difficulty of developing algorithms that can forge effective long-term relationships in many different scenarios. The results show that only S++ [43] was a top-performing algorithm across all metrics at all game lengths when associating with other algorithms. However, despite its fast learning speeds and its success in interacting with other machines in many different scenarios, S++ does not, in its current form, consistently forge cooperate relationships with people (SI.D), though it does cooperate with people as frequently as people cooperate with each other in the same studies. Thus, none of these existing algorithms establishes effective long-term relationships with both people and machines.

We hypothesized that S++'s inability to consistently learn to cooperate with people appears to be tied to its inability to generate and respond to costless signals. Humans are known for their ability to effectively coordinate on cooperative equilibria using costless signals called cheap talk [15, 16]. However, while signaling comes naturally to humans, the same cannot be said of sophisticated AI algorithms, such as machine-learning algorithms. To be useful, costless signals should be connected with behavioral processes. Unfortunately, most machine-learning algorithms have low-level internal representations that are often not easily expressed in terms of high-level behavior, especially in arbitrary scenarios. As such, it is not obvious how these algorithms can be used to generate and respond to costless signals at levels that people understand.

Fortuitously, unlike typical machine-learning algorithms, the internal structure of S++ provides a clear, high-level representation of the algorithm's dynamic strategy that can be described in terms of the dynamics of the underlying experts. Since each expert encodes a high-level philosophy, S++ could potentially be used to generate signals (i.e., cheap talk) that describe its intentionality. Speech acts from its partner can also be compared to its experts' philosophies to improve its expert-selection mechanism. In this way, S++ can be augmented with the ability to generate and respond to cheap talk. The resulting new algorithm, dubbed S# (pronounced 'S sharp'), is depicted in Figure 1 (see Methods and SI.C for details about the algorithm).

We conducted a series of three user studies (see SI.D–F for details) involving 220 participants, who played in a total of 472 games, to determine the ability of S# to forge cooperative relationships with people. Representative results are found in the final (culminating) study, in which participants played three representative repeated games (drawn from distinct payoff families; see SI.A.3) via a computer interface that hid the identity of their partner. In some conditions, players could engage in cheap talk by sending messages at the beginning of each round via the computer interface. Consistent with prior work investigating cheap talk in repeated games [16], messages were limited to the predetermined speech acts available to S#.

The proportion of mutual cooperation achieved by Human-Human, Human-S#, and S#-S# pairings are shown in Figures 2a-b. When cheap talk was not permitted, Human-Human and Human-S# pairings did not frequently result in cooperative relationships. However, across all three games, the presence of cheap talk doubled the proportion of mutual cooperation experienced by these two pairings. While S#'s speech profile was distinct from that of humans (Figure 2c), subjective, post-interaction assessments indicate that S# used cheap talk to promote cooperation as effectively as people (Figure 2d). In fact, many participants were unable to distinguish S# from a human player (Figure 2e). Together, these results illustrate that, across the games studied, the combined behavioral and signaling strategies of S# were as effective as those of human players.


(a) Compute a set E of experts $\{e_j\}$ from the game description
A
B
A
B
e_1 e_2 e_3 e_4 e_5 e_6 e_7
(b) Identify experts whose potential $\rho_j(t)$ meets aspiration level $\alpha(t)$
Potential $\rho_j(t)$
$\alpha(t)$
e_1 e_2 e_3 e_4 e_5 e_6 e_7
(c) Compute set $E_{\mathrm{cong}}(t)$ of experts congruent with incoming signal
e_1 e_2 e_3 e_4 e_5 e_6 e_7
(d) Prune experts
$E(t) = \{e_j \in E_{\mathrm{cong}}(t) : \rho_j(t) \geq \alpha(t)\}$
then use algorithm S to select an expert
S
meet aspiration
congruent
e_1 e_2 e_3 e_4 e_5 e_6 e_7
e_6
(e) Follow the selected expert for m rounds
Partner
Partner
A B
A B
E_6
E_6
…
Round t
Round t+m-1
(f) Update aspiration level:
$\alpha(t+m) = \alpha(t)\lambda^m + R(1-\lambda^m)$
Update each expert according to its own internal representation
e_1 e_2 e_3 e_4 e_5 e_6 e_7
(g) Speech-generation mechanism for expert e_6
Start
S_0 S_1 S_2 S_3 S_4 S_5 S_6 S_7 S_8
NUL→15+0
d→r({11,12})
d→r({11,12})
d→r({11,12})
s→5
s→5
s→5
s→ε
s→6
s→ε
g→r({11,12})+13
g→r({11,12})+13
NUL→15+0
p→14+8
p→14
p→ε
g→r({11,12})+13
Internal state
S_i
State transition
Input event → Output signal (speech)
Randomization
r(M): randomly pick message from set M

Signal	Text
0	Do as I say, or I'll punish you.
1	I accept your last proposal.
2	I don't accept your proposal.
3	That's not fair.
4	I don't trust you.
5	Excellent!
6	Sweet. We are getting rich.
7	Give me another chance.
8	Okay. I forgive you.
9	I'm changing my strategy.
10	We can both do better than this.
11	Curse you.
12	You betrayed me.
13	You will pay for this!
14	In your face!
15	Let's always play <action pair>.
16	This round, let's play <action pair>.
17	Don't play <action>.
18	Let's alternative between <action pair> and <action pair>.
ε	<empty>

Event	Explanation
s	Expert is *satisfied* with new payoff
f	Expert *forgives* other player
d	Partner *defected* against S#
g	Partner profited from defection (*guilty*)
p	Expert *punished* guilty partner
u	Expert failed to punish guilty partner
NUL	Auto transition; no input considered

Figure 1: An overview of S#, an algorithm that interweaves signaling capabilities into S++ [43]. **(a)** Prior to beginning the game, S# uses the description of the game to compute a set E of expert strategies. Each expert encodes a strategy or learning algorithm that defines behavior over all game states. **(b)** S# computes the potential, or highest expected utility, of each expert in E. The potentials are then compared to an aspiration level $\alpha(t)$, which encodes the average per-round payoff that the algorithm believes is achievable, to determine a set of experts that could potentially meet the agent's aspirations. **(c)** S# determines which experts carry out plans that are congruent with its partner's last proposed plan. **(d)** S# selects an expert (using algorithm S [45, 46]) from among those experts that both potentially meet its aspirations (step b) and are congruent with its partner's latest proposal (step c). If $E(t)$ is empty, S# selects its expert from among the set of experts that meet its aspiration level (step b). The currently selected expert generates signals based on its game-generic state machine (bottom). Given the current state of the expert and game events, the expert produces speech from a predetermined list of speech acts. **(e)** The machine follows the strategy dictated by the selected expert for m rounds of the repeated game. **(f)** The machine updates its aspiration level based on the average reward R it has received over the last m rounds of the game. The experts are also updated according to their own internal representations. The algorithm then returns to step b. The process repeats for the duration of the repeated game. Details are given in SI.C. Note that S++ is identical to S# except that S++ (1) replaces step c with $E_{\mathrm{cong}}(t) = E$, and (2) does not generate speech acts.

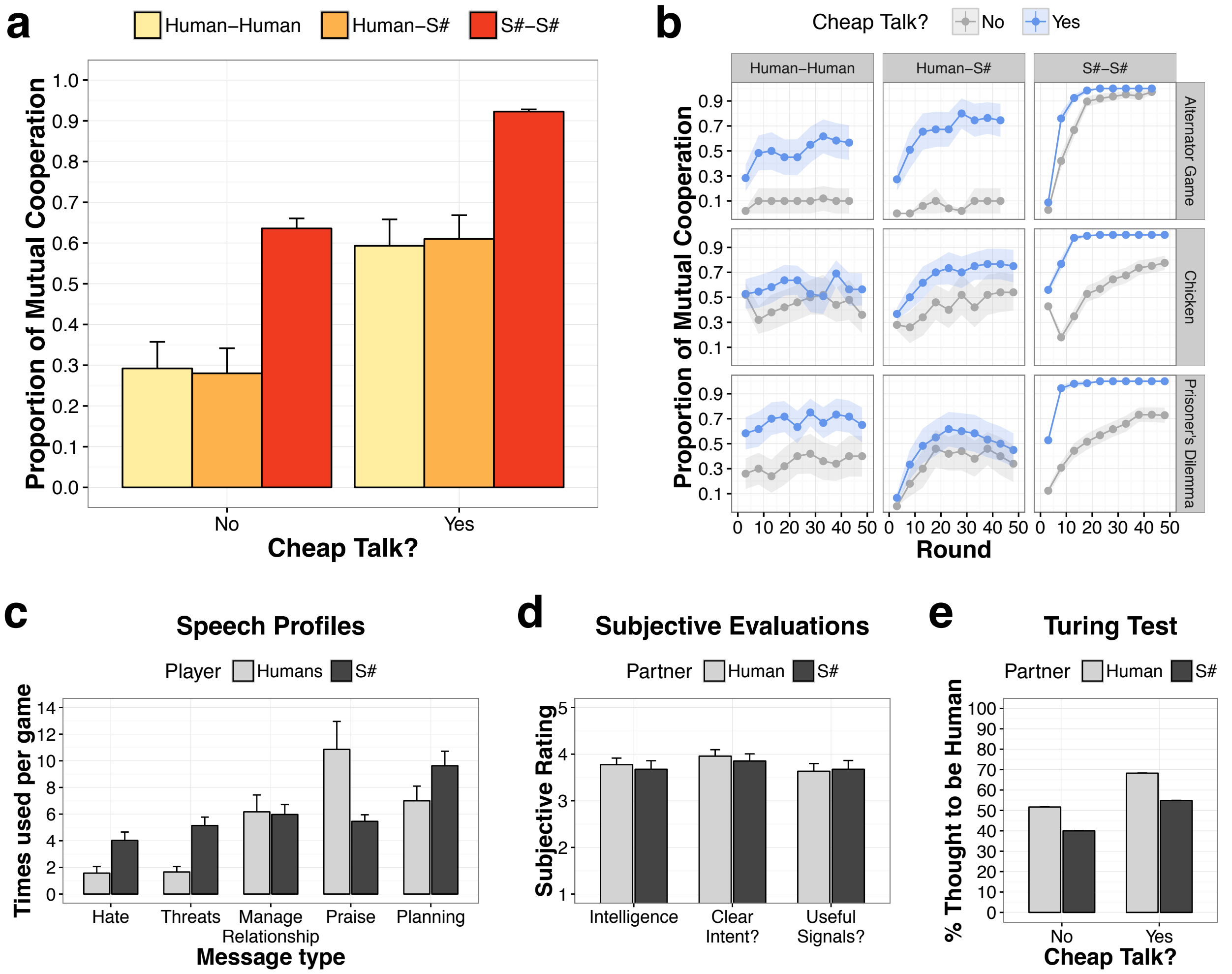


Figure 2: Results of the culminating user study in which 66 volunteer participants (people) were paired with each other and S# in three representative games (Chicken, Alternator Game, and Prisoner's Dilemma). S# is identical to S++ when cheap talk is not permitted. Bars and lines show average values over all trials, while error bars and ribbons show the standard error of the mean. Full details related to sample size and statistical tests are provided in SI.F. **(a)** The average proportion of mutual cooperation across all three games under conditions in which cheap talk between players was either permitted or not permitted. **(b)** The average proportion of mutual cooperation over time in each game in each pairing and condition. **(c)** The average number of times that Humans and S# used messages of each type over the course of an interaction when paired with people across all games. For simplicity, the 19 speech acts were grouped into five categories (see SI.F.1.3). S# tended to use more negative speech acts (labeled *Hate* and *Threats*), while people tended to use more positive speech acts (*praise*). **(d)** Results of three post-experiment questions for subjects that experienced the condition in which cheap talk was permitted. Participants rated (1) the intelligence of their partner, (2) the clarity of their partner's intentions, and (3) the usefulness of the communication between them and their partner. Answers were given on a 5-point Likert Scale. Specific questions and scales are provided in SI.F. **(e)** The percentage of time that human participants and S# were thought to be human by their partner when cheap talk was both permitted and not permitted.

2.2 Distinguishing Algorithmic Mechanisms

Why is S# so successful in forging cooperative relationships with both people and other algorithms? Are its algorithmic mechanisms fundamentally different from those of other algorithms for repeated games? We have identified three algorithmic mechanisms responsible for S#'s success. Clearly, Figure 2 demonstrates that the first of these mechanisms is S#'s ability to generate and respond to relevant signals people can understand, a trait not present in previous learning algorithms designed for repeated interactions. These signaling capabilities expand S#'s flexibility in that they also allow S# to more consistently forge cooperative relationships with people. Figure 3a demonstrates one simple reason that this mechanism is so important: signals help both S# and humans to more quickly experience mutual cooperation with their partners.

Second, our implementation of S# uses a rich set of experts that includes a variety of equilibrium strategies and even a simple learning algorithm (see SI.C.1). While none of these individual experts has an overly complex representation (e.g., no expert remembers the full history of play), these experts are more sophisticated than those traditionally considered (though not explicitly excluded) in the discussion of expert algorithms [22, 23, 24]. This more sophisticated set of experts permits S# to adapt to a variety of partners and game types, whereas algorithms that rely on a single strategy or a less sophisticated set of experts are only successful in particular kinds of games played with particular partners [25] (Figure 3c). Thus, in general, simplifying S# by removing experts from this set will tend to limit the algorithm's flexibility and generality, though doing so will not always negatively impact its performance when paired with particular associates in particular games.

Finally, S#'s somewhat non-conventional expert-selection mechanism (see Eq. 1) is central to its success. While techniques such as ε-greedy exploration (e.g., EEE) and regret-matching (e,g., Exp3) have permeated algorithm development in the AI community, S# instead uses an expert-selection mechanism closely aligned with recognition-primed decision making [26]. Given the same full, rich set of experts, more traditional expert-selection mechanisms establish effective relationships in far fewer scenarios than S# (Figure 3c). Figures 3a-b provide insights into why this is so. Compared to the other expert-selection mechanisms, S# has a greater combined ability to quickly establish a cooperative relationship with its partner (Figure 3a) and then to maintain it (Figure 3b), a condition brought about by S#'s tendency to not deviate from cooperation after mutual cooperation has been established (i.e., loyalty).

The loyalty brought about by S#'s expert-selection mechanism helps explain why S#-S# pairings substantially outperformed Human-Human pairings in our study (Figure 2a-b). S#'s superior performance can be attributed to two human tendencies. First, while S# did not typically deviate from cooperation after successive rounds of mutual cooperation (Figure 3b), many human players did. Almost universally, such deviations led to reduced payoffs to the deviator. Second, a sizable portion of our participants failed to keep some of their verbal commitments. On the other hand, since S#'s verbal commitments are derived from its intended future behavior, it typically carries out the plans it proposes. Had participants followed S#'s strategy in these two regards, Human-Human pairings would have performed nearly as well, on average, as S#-S# pairings (Figure 3d – see SI.F.4 for details).

2.3 Repeated Stochastic Games

The previous results were demonstrated for normal-form games. However, S++ also learns effectively in repeated stochastic games [27], which are more complex scenarios in which a round consists of a sequence of moves by both players. In these games, S++ is distinguished, again, by its ability to adapt to many different machine associates in a variety of different scenarios [27]. As in normal-form games, S++ can be augmented with cheap talk to form S#. While S++ does not consistently forge effective relationships with

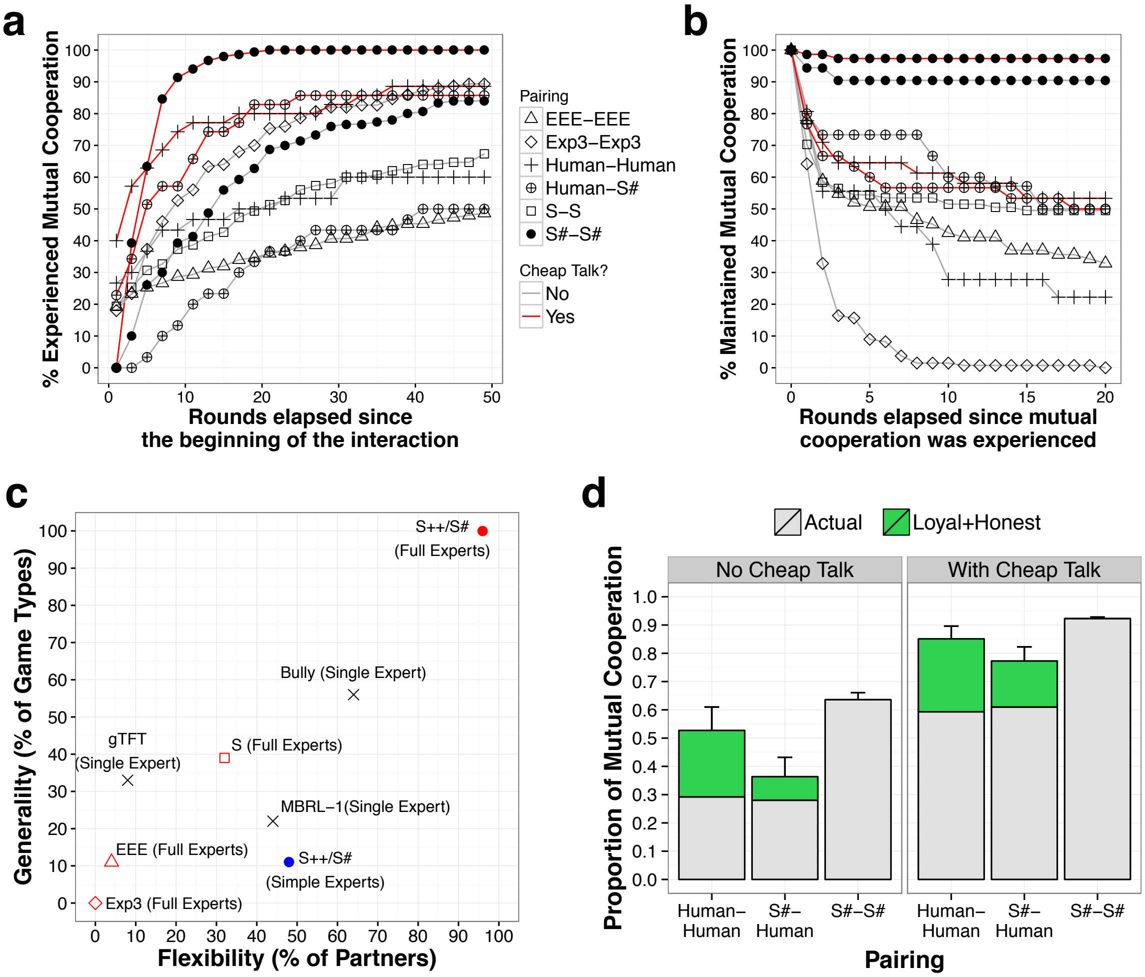


Figure 3: **(a)** Empirically generated cumulative distribution functions for the number of rounds required for pairings to experience two consecutive rounds of mutual cooperation across three different repeated games (Chicken, Alternator Game, and Prisoner's Dilemma). Per-game results are provided in SI.F. For machine-machine pairings, the results are obtained from 50 trials conducted in each game, whereas pairings with humans use results from a total of 36 different pairings each. **(b)** The percentage of partnerships for each pairing that did not deviate from mutual cooperation once the players experienced two consecutive rounds of mutual cooperation across the same three repeated games. **(c)** The percentage of game types (payoff family × game length) and partners (25 different algorithms) against which various algorithms were ranked in the top 2 (among the 25 different algorithms considered) with respect to payoffs received. See SI.B.5 for details. **(d)** The estimated proportion of rounds that would have resulted in mutual cooperation had all human players followed S#'s learned behavioral and signaling strategies of (1) not deviating from cooperative behavior when mutual cooperation was established (i.e., *Loyal*) and (2) following through with verbal commitments (i.e., *Honest*). See SI.F.4 for details. Error bars show the standard error of the mean. Had all human participants been loyal and honest, these results indicate that there would have been little difference between Human-Human and S#-S# pairings.

people in these more complex scenarios, our results show that S# does. Representative results are shown in Figure 4, which considers a turning-taking scenario in which two players must learn how to share a set of blocks. Like people, S# uses cheap talk to substantially increase its payoffs when associating with other people in this game (Figure 4b). These results mirror those we observe in normal-form games (compare Figures 4b and 2b). See SI.E for additional details and results.

3 Discussion

Our studies of human-S# partnerships were limited to five repeated games, selected carefully to represent different classes of games from the periodic table of games (see SI.A.3). Though future work should address more scenarios, S#'s success in establishing cooperative relationships with people in these representative games, along with its consistently high performance across all classes of 2x2 games and various repeated stochastic games [27] when associating with other algorithms, gives us some confidence that these results will generalize to many other scenarios.

Since Alan Turing envisioned Artificial Intelligence, major milestones have focused on defeating humans in zero-sum encounters [6, 7, 8, 9, 10]. However, in many scenarios, successful machines must cooperate with, rather than compete against, humans and other machines, even in the midst of conflicting interests and threats of being exploited. Our work demonstrates how autonomous machines can learn to establish cooperative relationships with people and other machines in repeated interactions. We showed that human-machine and machine-machine cooperation is achievable using a non-trivial, but ultimately simple, set of algorithmic mechanisms. These mechanisms include computing a variety of *expert* strategies optimized for various scenarios, a particular meta-strategy for a particular meta-strategy for *selecting experts to follow*, and the ability to generate and respond to simple *signals*. We hope that this first extensive demonstration of human cooperation with autonomous machines in repeated games will spur significant further research that will ensure that autonomous machines, designed to carry out human endeavors, will cooperate with humanity.

4 Methods

Detailed methods and analysis are provided in the SI. In this section, we overview three different aspects of these methods and analysis: the benchmark of games used to compare algorithms and people, results from our comparison of AI algorithms, and a description of S#.

4.1 Benchmark Games for Studying Cooperation

As with all historical grand challenges in AI, it is important to identify a class of benchmark problems to compare the performance of different algorithms. When it comes to human cooperation, a fundamental benchmark has been 2×2, general-sum, repeated games [28]. This class of games has been a workhorse for decades in the fields of behavioral economics [29], mathematical biology [30], psychology [31], sociology [32], computer science [33], and political science [34]. These fields have revealed many aspects of human cooperative behavior through canonical games, such as the Prisoner's Dilemmas, Chicken, Battle of the Sexes, and the Stag Hunt. Such games, therefore, provide a well-established, extensively studied, and widely understood benchmark for studying the capabilities of machines to develop cooperative relationships.

The periodic table of 2×2 games (Figure 5; see SI.A.3; [28, 35, 36, 37, 38]) identifies and categorizes 144 unique game structures that present many unique scenarios in which machines may need to cooperate.

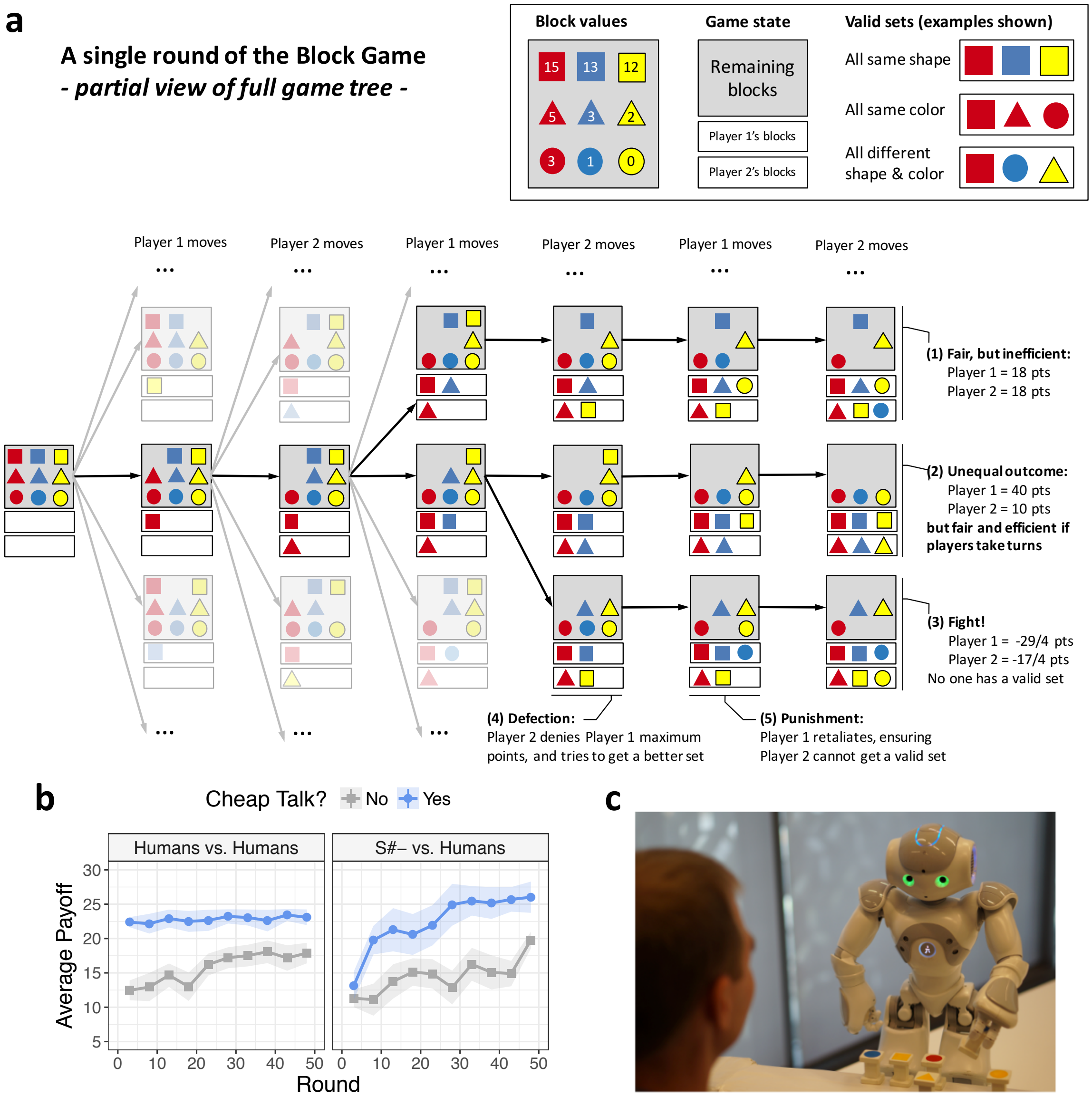
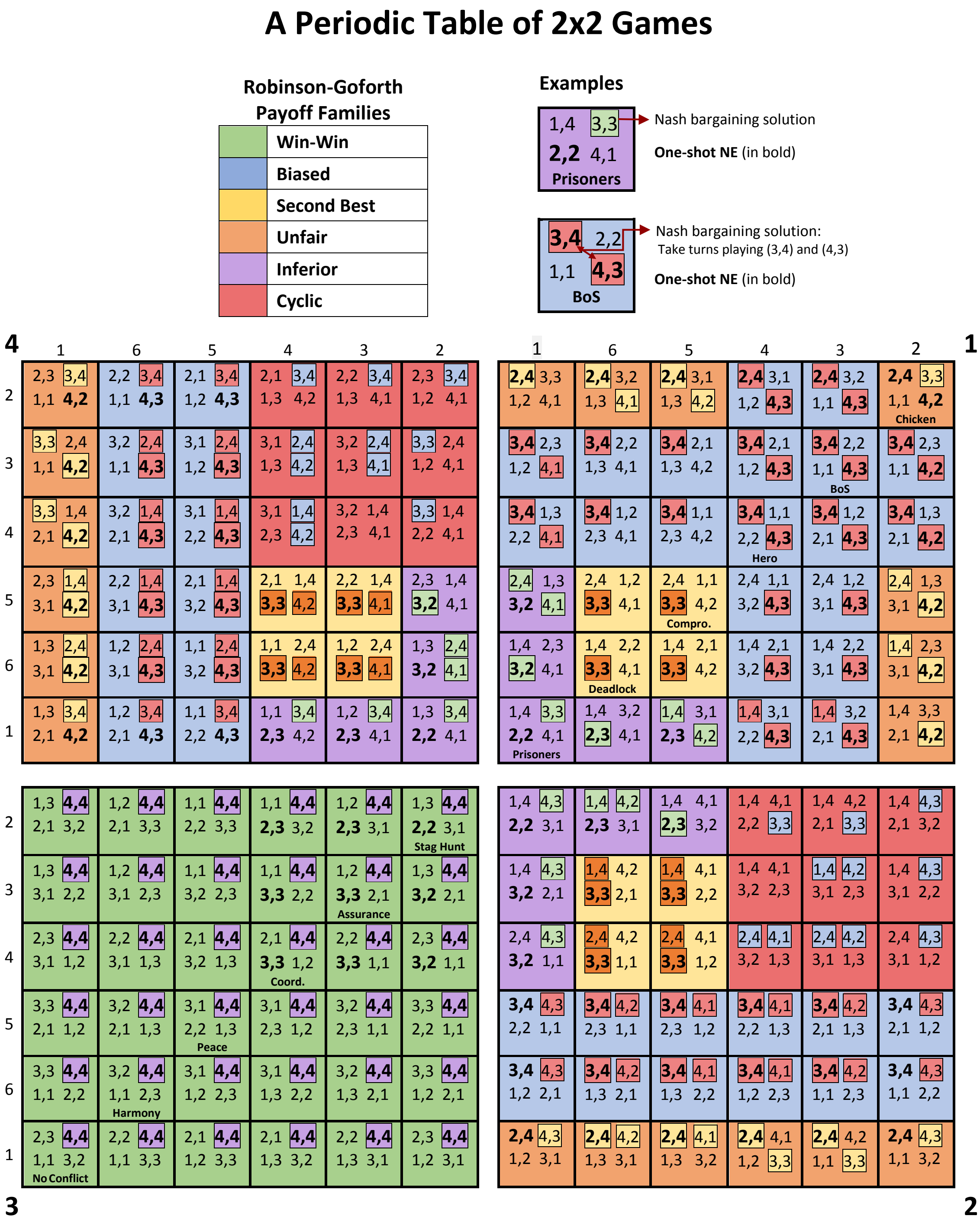


Figure 4: In addition to evaluating algorithms in normal-form games, we also evaluated people and algorithms in repeated stochastic games (including extensive-form games). Results are provided in SI.E. **(a)** An extensive-form game in which two players share a nine-piece block set. The two players take turns selecting blocks from the set until each has three blocks. The goal of each player is to get a *valid* set of blocks with the highest value possible, where the value of a set is determined by the sum of the numbers on the blocks. Invalid sets receive negative points. (1) A fair, but inefficient outcome in which both players receive 18 points. (2) An unequal outcome in which one player receives 40 points, while the other player receives just 10 points. However, when the players take turns getting the higher payoff (selecting all the squares), this is the Nash bargaining solution of the game, producing an average payoff of 25 to both players. (3) An outcome in which neither player obtains a valid set, and hence both players lose points. (4) This particular negative outcome is brought about when player 2 defects against player 1 by taking the block that player 1 needs to complete its (most-valuable) set. (5) Player 1 then retaliates to ensure that player 2 does not get a valid set either. **(b)** Average payoffs obtained by people and S#- (an early version of S# that generates, but does not respond to, cheap talk) when associating with people in the extensive-form game depicted in a. As in normal-form games, S#- successfully uses cheap talk to consistently forge cooperative relationships with people in this repeated stochastic game. For more details see SI.E. **(c)** We also implemented S#- on a Nao robot to play the Block Game with people.



A Periodic Table of 2x2 Games

Robinson-Goforth Payoff Families	
🟩	Win-Win
🟦	Biased
🟨	Second Best
🟧	Unfair
🟪	Inferior
🟥	Cyclic

Examples

1,4	3,3
2,2	4,1

Prisoners

Nash bargaining solution

One-shot NE (in bold)

3,4	2,2
1,1	**4,3**

BoS

Nash bargaining solution:
Take turns playing (3,4) and (4,3)

One-shot NE (in bold)

4	1	6	5	4	3	2
2	2,3 3,4 1,1 **4,2**	2,2 3,4 1,1 **4,3**	2,1 3,4 1,2 **4,3**	2,1 3,4 1,3 4,2	2,2 3,4 1,3 4,1	2,3 3,4 1,2 4,1
3	3,3 2,4 1,1 **4,2**	3,2 2,4 1,1 **4,3**	3,1 2,4 1,2 **4,3**	3,1 2,4 1,3 4,2	3,2 2,4 1,3 4,1	3,3 2,4 1,2 4,1
4	3,3 1,4 2,1 **4,2**	3,2 1,4 2,1 **4,3**	3,1 1,4 2,2 **4,3**	3,1 1,4 2,3 4,2	3,2 1,4 2,3 4,1	3,3 1,4 2,2 4,1
5	2,3 1,4 3,1 **4,2**	2,2 1,4 3,1 **4,3**	2,1 1,4 3,2 **4,3**	2,1 1,4 **3,3** 4,2	2,2 1,4 **3,3** 4,1	2,3 1,4 **3,2** 4,1
6	1,3 2,4 3,1 **4,2**	1,2 2,4 3,1 **4,3**	1,1 2,4 3,2 **4,3**	1,1 2,4 **3,3** 4,2	1,2 2,4 **3,3** 4,1	1,3 2,4 **3,2** 4,1
1	1,3 3,4 2,1 **4,2**	1,2 3,4 2,1 **4,3**	1,1 3,4 2,2 **4,3**	1,1 3,4 **2,3** 4,2	1,2 3,4 **2,3** 4,1	1,3 3,4 **2,2** 4,1

1	1	6	5	4	3	2
2	**2,4** 3,3 1,2 4,1	**2,4** 3,2 1,3 4,1	**2,4** 3,1 1,3 4,2	**2,4** 3,1 1,2 **4,3**	**2,4** 3,2 1,1 **4,3**	**2,4** 3,3 1,1 **4,2** Chicken
3	**3,4** 2,3 1,2 4,1	**3,4** 2,2 1,3 4,1	**3,4** 2,1 1,3 4,2	**3,4** 2,1 1,2 **4,3**	**3,4** 2,2 1,1 **4,3** BoS	**3,4** 2,3 1,1 **4,2**
4	**3,4** 1,3 2,2 4,1	**3,4** 1,2 2,3 4,1	**3,4** 1,1 2,3 4,2	**3,4** 1,1 2,2 **4,3** Hero	**3,4** 1,2 2,1 **4,3**	**3,4** 1,3 2,1 **4,2**
5	2,4 1,3 **3,2** 4,1	2,4 1,2 **3,3** 4,1	2,4 1,1 **3,3** 4,2 Compro.	2,4 1,1 3,2 **4,3**	2,4 1,2 3,1 **4,3**	2,4 1,3 3,1 **4,2**
6	1,4 2,3 **3,2** 4,1	1,4 2,2 **3,3** 4,1 Deadlock	1,4 2,1 **3,3** 4,2	1,4 2,1 3,2 **4,3**	1,4 2,2 3,1 **4,3**	1,4 2,3 3,1 **4,2**
1	1,4 3,3 **2,2** 4,1 Prisoners	1,4 3,2 **2,3** 4,1	1,4 3,1 **2,3** 4,2	1,4 3,1 2,2 **4,3**	1,4 3,2 2,1 **4,3**	1,4 3,3 2,1 **4,2**

3						
2	1,3 **4,4** 2,1 3,2	1,2 **4,4** 2,1 3,3	1,1 **4,4** 2,2 3,3	1,1 **4,4** **2,3** 3,2	1,2 **4,4** **2,3** 3,1	1,3 **4,4** **2,2** 3,1 Stag Hunt
3	1,3 **4,4** 3,1 2,2	1,2 **4,4** 3,1 2,3	1,1 **4,4** 3,2 2,3	1,1 **4,4** **3,3** 2,2	1,2 **4,4** **3,3** 2,1 Assurance	1,3 **4,4** **3,2** 2,1
4	2,3 **4,4** 3,1 1,2	2,2 **4,4** 3,1 1,3	2,1 **4,4** 3,2 1,3	2,1 **4,4** **3,3** 1,2 Coord.	2,2 **4,4** **3,3** 1,1	2,3 **4,4** **3,2** 1,1
5	3,3 **4,4** 2,1 1,2	3,2 **4,4** 2,1 1,3	3,1 **4,4** 2,2 1,3 Peace	3,1 **4,4** 2,3 1,2	3,2 **4,4** 2,3 1,1	3,3 **4,4** 2,2 1,1
6	3,3 **4,4** 1,1 2,2	3,2 **4,4** 1,1 2,3 Harmony	3,1 **4,4** 1,2 2,3	3,1 **4,4** 1,3 2,2	3,2 **4,4** 1,3 2,1	3,3 **4,4** 1,2 2,1
1	2,3 **4,4** 1,1 3,2 No Conflict	2,2 **4,4** 1,1 3,3	2,1 **4,4** 1,2 3,3	2,1 **4,4** 1,3 3,2	2,2 **4,4** 1,3 3,1	2,3 **4,4** 1,2 3,1

2						
	1,4 4,3 **2,2** 3,1	1,4 4,2 **2,3** 3,1	1,4 4,1 **2,3** 3,2	1,4 4,1 2,2 3,3	1,4 4,2 2,1 3,3	1,4 4,3 2,1 3,2
	1,4 4,3 **3,2** 2,1	1,4 4,2 **3,3** 2,1	1,4 4,1 **3,3** 2,2	1,4 4,1 3,2 2,3	1,4 4,2 3,1 2,3	1,4 4,3 3,1 2,2
	2,4 4,3 **3,2** 1,1	2,4 4,2 **3,3** 1,1	2,4 4,1 **3,3** 1,2	2,4 4,1 3,2 1,3	2,4 4,2 3,1 1,3	2,4 4,3 3,1 1,2
	3,4 4,3 2,2 1,1	**3,4** 4,2 2,3 1,1	**3,4** 4,1 2,3 1,2	**3,4** 4,1 2,2 1,3	**3,4** 4,2 2,1 1,3	**3,4** 4,3 2,1 1,2
	3,4 4,3 1,2 2,1	**3,4** 4,2 1,3 2,1	**3,4** 4,1 1,3 2,2	**3,4** 4,1 1,2 2,3	**3,4** 4,2 1,1 2,3	**3,4** 4,3 1,1 2,2
	2,4 4,3 1,2 3,1	**2,4** 4,2 1,3 3,1	**2,4** 4,1 1,3 3,2	**2,4** 4,1 1,2 3,3	**2,4** 4,2 1,1 3,3	**2,4** 4,3 1,1 3,2

Figure 5: We compared algorithms across the periodic table of 2x2 games based on the topology of Robinson and Goforth [37] for scenarios in which the players exhibit a strict ordinal preference ordering over the four game outcomes (specified by the values 1, 2, 3, and 4). For each game, the pure-strategy one-shot Nash equilibria (NEs) are given in bold-face type. The solutions played in the Nash bargaining solution (NBS [47] – i.e., the mutually cooperative solution) given the payoff values 1, 2, 3, and 4 are also highlighted, though the frequency at which each solution is played is not specified. Note that since the NBS depends on the actual payoffs and not just the preference ordering, other NBSs are possible for each game structure. The figure is adapted from the graphic developed by Bruns [38].

We use this set of game structures as a benchmark against which to compare the abilities of algorithms to cooperate. Successful algorithms should be able to forge successful relationships with both people and machines across all of these repeated games. In particular, we can use these games to quantify the abilities of various state-of-the-art machine learning algorithms to satisfy the aforementioned properties: generality across games, flexibility across opponent types (including humans), and speed of learning.

Like the majority of work in repeated interactions, we focus on two-player normal-form games to more easily understand how machines can forge cooperative relationships with people. Nevertheless, we are interested in algorithms that can also be used in more complex interactions, including the more general case of repeated (two-player) stochastic games (see, for example, Figure 4). Studies evaluating the ability of S# to forge cooperative relationships with people in repeated stochastic games have yielded similar results to those we report for two-player normal-form games (e.g., Figure 4b). These studies are described in SI.E.

4.2 Interacting with Other Machines: AI Algorithms for Repeated Interactions

With the goal of identifying successful algorithmic mechanisms for playing arbitrary repeated games, we selected and evaluated 25 existing algorithms (see Figure 6a) with respect to six different performance metrics (see SI.B.2) across the periodic table of 2x2 games. These representative algorithms included classical algorithms such as (generalized) generous tit-for-tat (i.e., GODFATHER) and win-stay-lose-shift (WSLS) [39], evolutionarily evolved memory-one and memory-two stochastic strategies [40], machine-learning algorithms (including reinforcement learning), belief-based algorithms [41], and expert algorithms [42, 43].

Results of this evaluation are summarized in Figure 6a. Detailed analysis is provided in SI.B. We make two high-level observations. First, it is interesting to observe which algorithms were less successful in these evaluations. For instance, while generalized tit-for-tat, WSLS, and memory-one and memory-two stochastic strategies (e.g., MEM-1 and MEM-2) are successful in prisoner's dilemmas, they are not consistently effective across the full set of 2x2 games. These algorithms are particularly ineffective in longer interactions, as they do not effectively adapt to their associate's behavior. Additionally, algorithms that minimize regret (e.g., Exp3 [22], GIGA-WoLF [23], and WMA [24]), which is the central component of world-champion computer poker algorithms [9], also performed poorly.

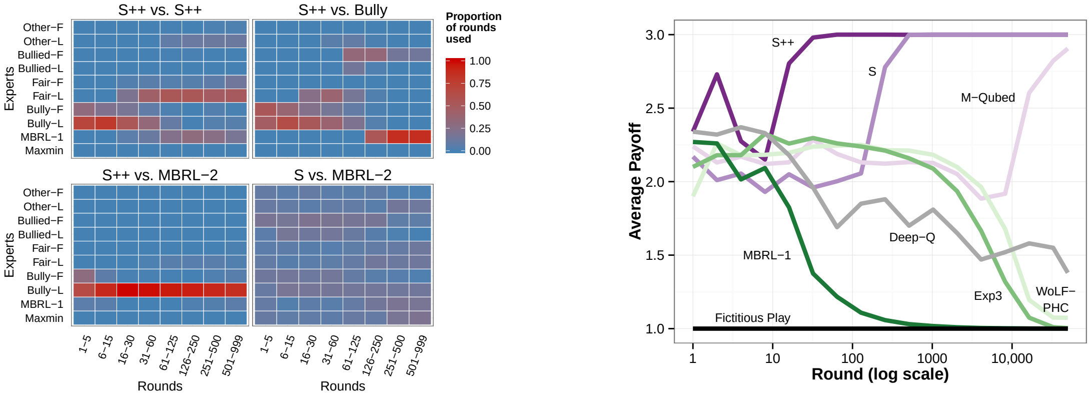
Second, while many algorithms had high performance with respect to some measure, only S++ [43] was a top-performing algorithm across all metrics at all game lengths. Furthermore, it maintained this high performance in each class of game and when associating with each class of algorithm (see SI.B.5). S++ learns to cooperate with like-minded associates, exploit weaker competition, and bound its worst-case performance (Figure 6b). Perhaps most importantly, whereas many machine-learning algorithms do not learn cooperative behavior until after thousands of rounds of interaction (if at all), S++ tends to do so within relatively few rounds of interaction (Figure 6c), likely fast enough to support interactions with people.

4.3 S#: A Machine-Learning Algorithm that Talks

S# is derived from S++ [43], an expert algorithm that combines and builds on decades of research in computer science, economics, and the behavioral and social sciences. S++ uses the description of the game environment to compute a diverse set of experts, each of which uses distinct mathematics and assumptions to produce a strategy over the entire space of the game. S++ then uses a meta-level control strategy based on aspiration learning [44, 45, 46] to dynamically reduce this set of experts. Formally, let E denote the set of experts computed by S++. In each epoch (beginning in round t), S++ computes the potential $\rho_j(t)$ of each expert $e_j \in E$, and compares this potential with its aspiration level $\alpha(t)$ to form a reduced set $E(t)$ of

Algorithm	Round-Robin Average	% Best Score	Worst-Case Score	Replicator Dynamic	Group-1 Tourney	Group-2 Tourney	Rank Summary Best – Mean – Worst
S++	1, 1, 1	2, 1, 2	1, 1, 1	1, 1, 1	1, 1, 2	1, 1, 1	1 – 1.2 – 2
MANIPULATOR	3, 2, 3	4, 3, 8	5, 2, 4	6, 4, 3	5, 3, 3	5, 2, 2	2 – 3.7 – 8
BULLY	3, 2, 1	3, 2, 1	7, 13, 20	7, 3, 2	6, 2, 1	6, 3, 5	1 – 4.8 – 20
S++/SIMPLE	5, 4, 4	8, 5, 9	4, 6, 10	10, 2, 6	8, 4, 6	9, 4, 6	2 – 6.1 – 10
S	5, 5, 8	6, 7, 10	3, 3, 8	5, 5, 8	7, 5, 9	7, 5, 9	3 – 6.4 – 10
FICT. PLAY	2, 8, 14	1, 6, 10	2, 8, 16	3, 12, 15	2, 8, 12	4, 9, 14	1 – 8.1 – 16
MBRL-1	6, 6, 10	5, 4, 7	8, 7, 14	11, 11, 13	9, 7, 10	8, 7, 10	4 – 8.5 – 14
EEE	11, 8, 7	14, 9, 5	9, 4, 2	14, 10, 9	13, 9, 8	13, 10, 8	2 – 9.1 – 14
MBRL-2	14, 5, 5	13, 8, 6	19, 5, 3	18, 9, 4	18, 6, 5	18, 6, 4	3 – 9.2 – 19
MEM-1	6, 9, 13	7, 10, 21	6, 9, 17	2, 6, 10	3, 10, 17	2, 8, 15	2 – 9.5 – 21
M-QUBED	14, 20, 4	15, 20, 3	15, 19, 5	17, 19, 5	17, 21, 4	16, 21, 3	3 – 13.2 – 21
MEM-2	9, 11, 20	9, 11, 22	13, 17, 22	4, 13, 19	4, 13, 25	3, 12, 20	3 – 13.7 – 25
MANIP-GF	11, 11, 21	12, 12, 19	12, 11, 19	9, 7, 20	12, 14, 20	11, 13, 21	7 – 14.2 – 21
WOLF-PHC	17, 11, 13	18, 14, 14	18, 14, 18	16, 14, 14	16, 11, 11	15, 11, 11	11 – 14.2 – 18
QL	17, 17, 7	19, 19, 4	17, 18, 7	19, 18, 7	19, 20, 7	19, 18, 7	4 – 14.4 – 20
GTFT (GODFATHER)	11, 14, 22	11, 15, 20	11, 16, 23	8, 8, 22	10, 16, 21	10, 15, 22	8 – 15.3 – 23
EEE/SIMPLE	20, 15, 11	20, 17, 12	20, 10, 9	20, 16, 11	24, 15, 14	20, 16, 13	9 – 15.7 – 24
EXP3	19, 23, 11	16, 23, 15	16, 23, 6	15, 23, 12	15, 25, 13	17, 25, 12	6 – 17.2 – 25
CJAL	24, 14, 14	25, 14, 13	24, 12, 15	24, 17, 16	20, 12, 16	22, 14, 16	12 – 17.3 – 25
WSLS	9, 17, 24	10, 16, 24	10, 20, 24	12, 20, 24	11, 17, 24	12, 17, 25	9 – 17.6 – 25
GIGA-WOLF	14, 19, 23	17, 18, 23	14, 15, 21	13, 15, 23	14, 18, 22	14, 19, 23	13 – 18.1 – 23
WMA	21, 21, 15	21, 21, 16	22, 21, 12	22, 21, 17	21, 19, 15	23, 20, 17	12 – 19.2 – 23
STOCH. FP	21, 21, 15	22, 22, 17	23, 22, 11	23, 22, 18	25, 24, 18	25, 22, 18	11 – 20.5 – 25
EXP3/SIMPLE	21, 24, 16	23, 24, 18	21, 24, 13	21, 24, 21	22, 22, 19	21, 23, 19	13 – 20.9 – 24
RANDOM	24, 25, 25	24, 25, 25	25, 25, 25	25, 25, 25	23, 23, 23	24, 24, 24	23 – 24.4 – 25

(a) Rankings of algorithms across six different metrics at three different game lengths



(b) Illustration of S++'s learning dynamics in Chicken

(c) Self play in a Prisoner's Dilemma

Figure 6: Selected results comparing the performance of representative algorithms across the periodic table of 2x2 games (Figure 5; see also SI.A.3) when paired with other algorithms. **(a)** The rankings of 25 algorithms with respect to six performance metrics (see SI.B.2). A lower rank indicates higher performance. For each metric, the algorithms are ranked in 100-round, 1000-round, and 50,000-round games, respectively. Example: the 3-tuple $3, 2, 1$ indicates the algorithm was ranked 3^{rd}, 2^{nd}, and 1^{st} in 100, 1000, and 50,000-round games, respectively. **(b)** An illustration of S++'s learning dynamics in Chicken. For ease of understanding, experts are categorized into groups (see SI.C). Top-left: When (unknowingly) paired with an agent that uses the same algorithm, S++ initially seeks to bully its associate, but then switches to fair, cooperative experts when attempts to exploit are unsuccessful. Top-right: When paired with BULLY, S++ learns the best response, which is to be bullied, achieved by playing MBRL-1, Bully-L, or Bully-F. Bottom-left: S++ quickly learns to play experts that bully MBRL-2. Bottom-right: On the other hand, algorithm S does not learn to consistently bully MBRL-2, showing that S++'s pruning rule (Eq. 1) enables it to teach MBRL-2 to accept being bullied, thus producing high payoffs for S++. These results are averaged over 50 trials each. **(c)** The average per-round payoffs of various machine-learning algorithms over time in self play in a traditional (0-1-3-5)-Prisoner's Dilemma in which mutual cooperation produces a payoff of 3 and mutual defection produces a payoff of 1. Results are the averages of 50 trials. Among the machine-learning algorithms we evaluated, S++ is unique in its ability to quickly form successful relationships with other algorithms across the set of 2x2 games.

experts:

$$E(t) = \{e_j \in E : \rho_j(t) \geq \alpha(t)\}. \tag{1}$$

This reduced set consists of the experts that S++ believes could potentially produce satisfactory payoffs. It then selects one expert $e_{\text{sel}}(t) \in E(t)$ using a satisficing decision rule [45, 46]. Over the next m rounds, it follows the strategy prescribed by $e_{\text{sel}}(t)$, after which it updates its aspiration level as follows:

$$\alpha(t+m) \leftarrow \lambda^m \alpha(t) + (1 - \lambda^m) R, \tag{2}$$

where $\lambda \in (0, 1)$ is the learning rate and R is the average payoff obtained by S++ in the last m rounds. It also updates each expert $e_j \in E$ based on its peculiar reasoning mechanism, and then begins a new epoch.

These results demonstrate the ability of S++ to effectively establish and maintain profitable long-term relationships with machines in arbitrary repeated games. Does S++ also learn to form cooperative relationships with people?

S# differs from S++ in two ways. First, the partner's proposed plans, signaled via speech acts, are used to further reduce the set of experts that S# considers selecting (Figure 1c). Formally, let $E_{\text{cong}}(t)$ denote the set of experts in round t that are *congruent* with the last joint plan proposed by S#'s partner (see SI.C.2.2). Then, S# considers selecting experts from the following set:

$$E(t) = \{e_j \in E_{\text{cong}}(t) : \rho_j(t) \geq \alpha(t)\}. \tag{3}$$

If this set is empty (i.e., no desirable options are congruent with the partner's proposal), $E(t)$ is calculated as in Eq. (1). Second, S# also extends S++ by generating speech acts that convey the "stream of consciousness" of the algorithm (Figure 1d). Specifically, a finite-state machine with output is generated for each expert. Given the state of the expert and the game outcomes, the state machine of the currently selected expert produces speech derived from a predetermined set of phrases. The set of speech acts, which are largely game-generic (though some adaptations must be made for multi-stage games; see SI.E.3.4) allows S# to provide feedback to its partner, make threats, provide various explanations to manage the relationship, and propose and agree to plans.

See SI.C for an in-depth description of S#.

References

[1] A. M. Turing. Computing machinery and intelligence. *Mind*, pages 433–460, 1950.

[2] A. J. Toole, P. J. Phillips, F. Jiang, J. Ayyad, N. Penard, and H. Abdi. Face recognition algorithms surpass humans matching faces over changes in illumination. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(9):1642–1646, 2007.

[3] W. Youyou, M. Kosinski, and D. Stillwell. Computer-based personality judgments are more accurate than those made by humans. *Proceedings of the National Academy of Sciences*, 112(4):1036–1040, 2015.

[4] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, and S. Thrun. Junior: The Stanford entry in the urban challenge. *Journal of Field Robotics*, 25(9):569–597, 2008.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[6] M. Campbell, A. J. Hoane, and F. Hsu. Deep blue. *Artificial intelligence*, 134(1):57–83, 2002.

[7] J. Schaeffer, N. Burch, Y. Björnsson, A. Kishimoto, M. Müller, R. Lake, P. Lu, and S. Sutphen. Checkers is solved. *Science*, 317(5844):1518–1522, 2007.

[8] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, et al. Building Watson: An overview of the DeepQA project. *AI Magazine*, 31(3):59–79, 2010.

[9] M. Bowling, N. Burch, M. Johanson, and O. Tammelin. Heads-up limit holdem poker is solved. *Science*, 347(6218):145–149, 2015.

[10] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Angonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

[11] D. G. Rand, A. Peysakhovich, G. T. Kraft-Todd, G. E. Newman, O. Wurzbacher, M. A. Nowak, and J. D. Greene. Social heuristics shape intuitive cooperation. *Nature Communications*, 5, 2014.

[12] R. Boyd and P. J. Richerson. Culture and the evolution of human cooperation. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1533):3281–3288, 2009.

[13] R. H. Frank. *Passions Within Reason: The Strategic Role of the Emotions.* W. W. Norton & Company, 1988.

[14] B. Skyrms. *The Stag Hunt and the Evolution of Social Structure*. Cambridge Press, 2003.

[15] D. Sally. Conversation and cooperation in social dilemmas a meta-analysis of experiments from 1958 to 1992. *Rationality and society*, 7(1):58–92, 1995.

[16] D. Balliet. Communication and cooperation in social dilemmas: A meta-analytic review. *Rationality and society*, 54(1):39?57, 2009.

[17] A. Peysakhovich, M. A Nowak, and D. G. Rand. Humans display a cooperative phenotype that is domain general and temporally stable. *Nature Communications*, 5, 2014.

[18] G. Klein, P. J. Feltovich, J. M. Bradshaw, and D. D. Woods. Common ground and coordination in joint activity. *Organizational simulation*, 53, 2005.

[19] K. Dautenhahn. Socially intelligent robots: Dimensions of human–robot interaction. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 362(1480):679–704, 2007.

[20] C. Breazeal. Toward sociable robots. *Robotics and autonomous systems*, 42(3):167–175, 2003.

[21] E. Kamar, Y. Gal, and B. J. Grosz. Modeling information exchange opportunities for effective human–computer teamwork. *Artificial Intelligence*, 195:528–550, 2013.

[22] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: the adversarial multi-armed bandit problem. In *Proceedings of the 36th Symposium on the Foundations of Computer Science*, pages 322–331, 1995.

[23] M. Bowling. Convergence and no-regret in multiagent learning. In *Advances in Neural Information Processing Systems 17*, pages 209–216, 2004.

[24] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the 2nd European Conference on Computational Learning Theory*, pages 23–37, 1995.

[25] L. Marstaller, A. Hintze, and C. Adami. The evolution of representation in simple cognitive networks. *Neural Computation*, 25:2079–2107, 2013.

[26] G. Klein. Recognition-primed decisions. In W. B. Rouse, editor, *Advances in man-machine systems research*, volume 5, pages 47–92. Greenwhich, CT: JAI Press, 1989.

[27] J. W. Crandall. Robust learning in repeated stochastic games using meta-gaming. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2015. To appear.

[28] A. Rapoport and M. J. Guyer. *A Taxonomy of 2x2 Games*. Bobbs-Merrill, 1967.

[29] Colin Camerer. *Behavioral game theory*. New Age International, 2010.

[30] Josef Hofbauer and Karl Sigmund. *Evolutionary games and population dynamics*. Cambridge university press, 1998.

[31] D. G. Rand and M. A. Nowak. Human cooperation. *Trends in Cognitive Sciences*, 17(8):413–425, 2013.

[32] Peter Kollock. Social dilemmas: The anatomy of cooperation. *Annual review of sociology*, pages 183–214, 1998.

[33] M. L. Littman and P. Stone. A polynomial-time Nash equilibrium algorithm for repeated games. *Decision Support Systems*, 39:55–66, 2005.

[34] R. Axelrod. *The Evolution of Cooperation*. Basic Books, New York, 1984.

[35] A. Rapoport, M. J. Guyer, and D. G. Gordon. *The 2x2 Game*. The University of Michigan Press, 1976.

[36] S. J. Brams. *A Theory of Moves*. Cambridge University Press, 1994.

[37] D. Robinson and D. Goforth. *The Topology of the 2x2 Games: A New Period Table*. Routledge, 2005.

[38] B. Bruns. Navigating the topology of 2x2 games: An introductory note on payoff families, normalization, and natural order. *CoRR*, abs/1010.4727, 2010.

[39] M. Nowak and K. Sigmund. A strategy of win-stay, lose-shift that outperforms tit-for-tat in the prisoner's dilemma game. *Nature*, 364:56–58, 1993.

[40] D. Iliopoulous, A. Hintze, and C. Adami. Critical dynamics in the evolution of stochastic strategies for the iterated prisoner's dilemma. *PLOS Computational Biology*, 6(10):1–8, 2010.

[41] D. Fudenberg and D. K. Levine. *The Theory of Learning in Games*. The MIT Press, 1998.

[42] D. de Farias and N. Megiddo. Exploration–exploitation tradeoffs for expert algorithms in reactive environments. In *Advances in Neural Information Processing Systems 17*, pages 409–416, 2004.

[43] J. W. Crandall. Towards minimizing disappointment in repeated games. *Journal of Artificial Intelligence Research*, 49:111–142, 2014.

[44] H. A. Simon. Rational choice and the structure of the environment. *Psychological Review*, 63(2):129–138, 1956.

[45] R. Karandikar, D. Mookherjee, D. R., and F. Vega-Redondo. Evolving aspirations and cooperation. *Journal of Economic Theory*, 80:292–331, 1998.

[46] J. R. Stimpson, M. A. Goodrich, and L. C. Walters. Satisficing and learning cooperation in the prisoner's dilemma. In *Proceedings of the 17th National Conference on Artificial Intelligence*, pages 535–544, 2001.

[47] J. F. Nash. The bargaining problem. *Econometrica*, 28:155–162, 1950.

The supplementary information (SI) for this paper is available here.