A Multiagent Turing Test Based on a Prediction Market

Joseph Farfel
Dept. of Computer Science, Duke U.
Durham, NC 27708, USA
joseph.farfel@gmail.com

Vincent Conitzer
Dept. of Computer Science, Duke U.
Durham, NC 27708, USA
conitzer@cs.duke.edu

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems; J.4 [**Social and Behavioral Sciences**]: Economics

General Terms

Economics

Keywords

prediction markets, Turing tests, games with a purpose, deployed web-based applications, using points as an artificial currency

1. INTRODUCTION

In a Turing test, a single agent (the *judge*, generally a human) chats with two mysterious conversation partners [1]. One of the two mystery conversationalists is a human, while the other is a computer program (a *chat bot*, or just *bot*). The bot is the entity who is actually taking the test: If the judge cannot (accurately) tell which mystery conversation partner is the human and which is the bot, then the bot passes the test (and otherwise it fails). It is easy to see that a Turing test can also be run with only a single mysterious conversation partner (whom we will call the *target*). To do so, the test organizer chooses a human target with probability 50% and a computer target with probability 50%. Then, after a conversation with the target, the judge is asked to report how probable she thinks it is that the target is a human—if she reports 50% or higher when talking to a bot, then that bot passes the test.

One might imagine a variant of the Turing test where the "judge" consists of a *group* of agents (possibly humans), and the result of the test hinges upon the group's aggregate belief of the probability that the target is a human. This setting involves multiple judging agents, where each agent has her own belief, but where information about beliefs might be exchanged among agents during the course of the test. An agent's personal belief is updated throughout the test based on the information she receives about other agents' beliefs, and, of course, on the target's contributions to the conversation.

Our new web game, Turing Trade, is an implementation of a Turing test with a group as the judge—a true *multiagent* Turing test. In Turing Trade, a group of agents converses with a single target. Each individual agent in the group gets to ask the target public questions, and the target gives public answers. During the conversation, all individuals in the group are encouraged to competitively bet on the target's humanity, by buying and selling securities (with points, not real money). The price of these securities varies based on judges' bets, and at any given time in the game, this price is a measure of the group's consensus belief that the target is a human. At the end of the game, the target's true nature (human or computer) is revealed, and based on this some of the securities pay out. The betting part of the game is a *prediction market* [2], where the single binary event that the judges are trying to predict is "the target will be revealed to be a human."

Turing Trade can be played online at http://www.turingtrade.org. All logs from played games are posted publicly on the website. We believe the game offers numerous advantages over standard Turing test websites, including the promise of collecting significantly more data, and more finely grained data, for chat bot designers and others. The data is more finely grained because the judges are not providing just a final assessment of whether the target is a human, but rather a probability that changes over time. The Turing Trade project has additional purposes, including the creation of a novel, fast-paced prediction market that may provide useful lessons for the design of prediction markets in general. Another purpose is simply to create entertainment value for its players. If the game ends up being played by very many people, then, for example, providers of free e-mail accounts could use our game to ensure that bots do not sign up for accounts (a trick commonly used by spammers), by requiring a new user to play as the target in Turing Trade (and be judged human).

Cite as: A Multiagent Turing Test Based on a Prediction Market, Joseph Farfel, Vincent Conitzer, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009),* Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 1407 – 1408

2. GAME OVERVIEW

In a game of Turing Trade, the *target* is the single player (possibly a bot) whose humanity is being judged. The group judging the target is composed of n agents, called *judges* or *bettors*. The bettors ask the target questions, and bet on whether the target is a human or a computer. The target answers questions from the bettors, and tries to seem as human as possible, whether or not it is really a human.

2.1 Bot Targets

It is very important for our game to have a strong lineup of bots available to serve as targets. All the bots described below (except for Simple Bot) were written by others, and reside on their owners' web servers (it is not our intention to create new bots ourselves). When a game is in progress, the Turing Trade server initiates a new conversation with a bot, and simply sends it bettors' questions and receives the bot's answers. The current incarnation of the game features six different bots organized into three classes:

1. **Simple Bot.** This is a very simple bot—to any question, it replies with the same answer ("Hmmm...That's an interesting question.").

2. **Alice and iGod.** These bots are based on AIML, or the Artificial Intelligence Markup Language. AIML and Alice, the first bot to use it, are creations of Dr. Richard Wallace (http://www.alicebot.org); they are extensions of the logic underlying the classic bot Eliza, developed by Joseph Weizenbaum in 1966. The iGod bot is currently the most popular bot at the free AIML-bot hosting web site Pandorabots (http://www.pandorabots.com/). Alice won the Loebner Prize for most human-like chat bot in 2000, 2001, and 2004.

3. **Jabberwacky, George, and Joan.** These three bots are all based on Jabberwacky, by Rollo Carpenter (http://www.jabberwacky.com/). Its approach is heavily centered on learning, and it operates primarily by storing everything that any human has ever said to it, and using contextual pattern matching to find things in this vast database to say in response to new human input. The Jabberwacky bots George and Joan won the Loebner prize in 2005 and 2006.

2.2 Questions and Answers

The target always sees only one question at a time from the group of bettors. This question is called the *current* question. The target considers the current question, and sends its answer to the server; at this point, the server may send the target another single (current) question. The target may only send one answer for each question. From the target's perspective, the conversation is a simple back-and-forth exchange. The only indication the target has that it is talking to a group of people rather than a single person is that questions belonging to bettor i are tagged as such. This allows chat bots that currently exist to play Turing Trade unmodified.

Every bettor also gets to see the current question, at the same time as the target. Unlike the target, however, which may only submit an answer if there is an unanswered current question, any of the n bettors may submit a question to the game server at any point during the game. The server keeps a queue of questions, Q_i, for each player i, and initially, all question queues are empty. When the server receives a question, q, from a bettor i, it does the following:

- If there is no current (unanswered) question, broadcast question q to all bettors and the target. Question q is now the current question.
- Otherwise (there is a current question), add the question q to queue Q_i.

With this setup, all bettors and the target see the current question, but every other unanswered question is invisible to everyone but the bettor who asked the question. When the server receives an answer from the target to a current question q, it does the following:

1. Let i be the bettor who asked the question being answered (question q). Send the answer to bettor i, and send a signal (not containing the text of the answer) to every bettor $b \neq i$ signifying only that an answer to the question has been given.

2. Starting with $j = i+1 \pmod n$, and incrementing $j \pmod n$ after each check, search for the first nonempty queue Q_x.

   - If all queues are empty, do nothing (except wait for a bettor to send a question).
   - Otherwise (Q_x is the first nonempty queue), remove the first question q' from Q_x. This is the new current question; send q' to all bettors and the target.

3. Five seconds after step 1 (sending the answer to question q to player i), send the text of the answer (to q) to every bettor $b \neq i$.

This scheme ensures that questions are taken from bettors in a round-robin manner, unless some bettors are not asking questions, in which case they are skipped. We note that bettor i gets to see the answer to her question five seconds earlier than every bettor $b \neq i$. This delay rewards bettors for asking good questions (where a good question is one that reveals a lot about the nature of the target), because it allows the bettor who asked the question to trade on this information before it becomes available to the other bettors.

2.3 Betting

At any time during the conversation with the target, any bettor may place a bet on whether or not the target is human. A bet is made by buying or selling a *human security* or a *computer security*. A human security is an asset of the form "Pays 100 points if the target is revealed to be a human," while a computer security is an asset of the form "Pays 100 points if the target is revealed to be a computer." Securities pay out at the end of the game, when the target's nature is revealed to bettors: for example, if the target is a human, a human security pays out 100 and a computer security pays out 0. Since the two types of security are complementary (owning one of each type of security is equivalent to owning 100 points), we without loss of generality restrict every bettor to own at most one type of security at a time.

Human and computer securities are bought from and sold to a central *market maker*, who has an infinite supply of securities to sell, and an infinite willingness to buy securities. The market maker always sets the price for the computer security at 100 minus the price of the human security (this is ignoring a small bid-ask spread that we will discuss shortly). A bettor can purchase or sell one security at a time. When a human security is purchased (or a computer security is sold), the price for human securities increases by 1, and when a human security is sold (or a computer security is purchased) the price for human securitites decreases by 1.

The market maker maintains a spread of 1 between bid and ask prices. This is done to prevent arbitrage: with the spread, a bettor can buy a security for the ask price of x from the market maker (causing the ask price to increase to $x + 1$), and then sell it back for the bid price $(x + 1) - 1 = x$. Neither the bettor nor the market maker profits if this happens, but without the spread, the bettor would have had a profit of 1. The maximum price for a human security is 100, and the minimum price is 0.

The price to buy a human security is plotted over the course of a game. Local equilibria in the price measure the bettors' consensus belief (at some point in time) of how probable it is that the target will be revealed to be a human. For example, if the human security price is hovering around 70, then the bettors, in aggregate, believe that the there is a 70 percent chance that the target will be revealed to be a human at the end of the game. This interpretation relies upon the assumption that the bettors are rational (in the sense of maximizing their expected number of points), and upon the fact that a prediction market such as this one offers incentives for rational bettors to update the consensus probability in ways consistent with their true beliefs (at least for a myopic sense of rationality).

Acknowledgments: Supported by NSF IIS-0812113, the Sloan Foundation, and a Yahoo! Faculty Research Grant.

3. REFERENCES

[1] A. Turing. Computing machinery and intelligence. *Mind*, 59, 1950.

[2] J. Wolfers and E. Zitzewitz. Prediction Markets. *The Journal of Economic Perspectives*, 18(2):107–126, 2004.