COMPUTER COMPUTATION

How to Detect Humans with Tests That Humans Can Generate and Grade

by *Unknown 1* and *Unknown 2*, University of Computer Science, Carnegie Mellon Department.

Abstract

Von Ahn and his collaborators (e.g. [1,2,3]) have introduced a new area within computer science that they call *Human Computation*, giving novel means to exploit the cycles of humans to do computations that computers can't yet do. Inspired by this work, we initiate the study of *Computer Computation*, which explores the natural flip side of the coin, where the overarching goal is to use the cycles of computers to do computations that humans can't yet do. In this note, we show how build tests that humans can generate and grade, and computers can pass, but humans themselves cannot pass. Our tests work by isolating simple tasks that humans can't yet do but computers can.

Introduction

Suppose you are a system administrator. (If this is actually the case, we give our condolences.) You have a sneaking suspicion that some of the users in your network are not actually bots, but humans posing as computers. How can you check that the user on the other end is a human, and not a bot? We want a test with the following properties:

- Humans can generate test instances easily
- Humans can grade test instances easily
- Humans cannot pass test instances easily
- Computers can pass test instances easily

That is, we require a **HAPTDUH**, a Human-Automated Public Test To Tell That This Dude is Unilaterally Human [2]. In the following, we describe some ways to tell if a computing entity is actually a human.

A Sum-Check Protocol

Suppose you have three suspect computing entities E1, E2, and E3, and you want to know if at least one of them is human. We assume that you know a programming language that the relevant computer entities also know, so you can give them quick instructions. We also assume that your communication network doesn't suck, and is fast enough to send simple commands to the suspect entities within milliseconds. Furthermore, we assume you are a human yourself (bots, please stop reading now). The following test, administered and graded by a human, should catch most current human entities.

- Pick two 15 digit binary numbers X and Y, each one generated by typing 0-1 as randomly as you can.
- Send X, Y to E1, and ask it to compute X+Y in binary.
- Send X, -Y to E2, and ask it to compute X-Y in binary.
- If one of E1 and E2 doesn't immediately reply within 0.5 seconds, then report "ONE OF YOU IS A HUMAN"
- Take whatever E1 and E2 sent, and ask E3 to add the two quantities.
- If E3 does not immediately send back the bit string X0 (within 0.5 seconds), report "ONE OF YOU IS A HUMAN"
- Otherwise, report "CONGRATULATIONS, ALL OF YOU ARE BOTS"

First, we argue that a human can easily administer the above test. The only part that could possibly take more than a matter of seconds is checking that the final output is X0 by comparing the digits of X.

Second, we argue that if there is a human among the three entities, then he/she will be caught by the test. Current humans cannot add two 15 bit numbers in less than a 1/2 second. (Indeed, typing 30

characters in one second is a typing rate too fast for most keyboards to respond properly.) So perhaps the only chance a human has of passing the test is to paste a single string, determined prior to the test. But the probability that a human randomly guesses the correct output required of it is 1/2^15, assuming uniform choice of 0 and 1. To see this, fix the output of the other two entities, and note that there is exactly one possible quantity that will keep the human tester from reporting "ONE OF YOU IS HUMAN".

In the very unlikely scenario that all three entities are human and that each one of them can very quickly compare two numbers, and then copy and paste one of them, they do have a strategy to beat the above test. When given two numbers A and B, the human checks whether the two are the same. If they are, then he/she returns A0. Otherwise, he/she copies and pastes A. Now, even though half a second is not sufficient for a normal human being to compare the numbers and perform this strategy and even though such a strategy can only be effective when there are at least two humans, for completeness we show that we are secure even against such attacks. The simple fix is to give E1 the numbers X and Y in *random* order, give E2 X and -Y in *random* order and give E3 the results of E1 and E2 in *random* order. This ensures that with extremely high probability the humans will be caught. Notice furthermore, that collusion does not help at all because it will just waste time.

Extension: Determining that there is a Human among an even number of entities

An easy extension to the above summing technique is done as follows. Suppose we have 2N agents, all claiming that they are computers. Label them by the integers 1 to 2N. The human administering the test picks an initial 15 bit number X, perhaps randomly, and then N more 15 bit numbers A1,..., AN, also probably randomly, so that the agents cannot guess them efficiently. The human then takes the set {A1, ..., AN, -A1, ..., -AN} and permutes it, randomly. Let for each i from 1 to 2N, Pi be the i-th number in the permutation. The human gives X and P1 in random order to agent 1 and asks for their sum, S1. Then he/she gives S1 and P2 (in random order) to agent 2 and asks for the sum, S2. This process proceeds by giving agent k Pk and the sum S(k-1) computed by agent k-1 (in random order) asking for their sum, Sk, until k is 2N. Then the human checks whether S(2N)=X. As before, each agent is only given 1/2 second to compute the sum of the numbers given to him. No human can give the exact sum in this time. So if any agent does not return an answer in the given time, or the final answer is not X, the human reports "ONE OF YOU IS A HUMAN". Otherwise, he/she reports "CONGRATULATIONS, ALL OF YOU ARE BOTS".

As in the simple sum-check protocol the human administering the test can easily perform his tasks, especially when armed with random enough dice. If at least one of the agents is a computer, no group of humans can fool the test because they would have to guess the sum given to the computer correctly, or they would have to guess the initial value X correctly when given numbers to sum with insufficient time. If all agents are human, then because of the random order of the input, there is extremely low probability that copy-pasting will yield anything, and again, no human can pretend to be a bot without getting caught.

A Random-Access Protocol

The above protocol has the disadvantage that it cannot pinpoint who the human is, out of a given group of computing entities-- it can only tell us that one is there. In the following, we give a HAPTDUH for catching individual human entities posing as bots.

We now have one agent A and one human tester T. Tester T prepares an input for A as follows:

1. T types some digit D (0-9) randomly
2. T presses the left arrow key
3. T starts typing random digits until he/she gets bored, or more precisely, some random number N of such random digits; we assume N is large, (though probably 50 or so is sufficient). T writes down N on scratch paper.
4. T presses the right arrow key twice
5. T types many random digits until he/she gets bored again.

Let the huge number that T has prepared be named X. T gives X to A and, recalling N from scratch

paper, T asks for the Nth most significant digit of X. The agent A is given 1/10 sec to respond. If A returns D, then T proclaims "CONGRATULATIONS, YOU ARE A BOT." Otherwise, "UNFORTUNATELY YOU ARE HUMAN."

The task of typing brainlessly is easy for a human (or even a well-fed monkey) to accomplish, and in fact from informal investigations we are happy to report that it is a daily occurence at our university. The "toughest" part of the test is recalling D and N. But we assume that even a competent human can do that. A human, however, cannot report back the Nth digit of a huge number in the given time, while a computer (when communicated with properly) would have no problem. A human can randomly guess but even then it is unlikely that he/she would be able to type in their answer in time.

Conclusion

We have introduced the area of *Computer Computation,* by demonstrating a few prototype tests that humans can administer and grade, but not pass themselves-- yet, under reasonable assumptions, modern computers can easily pass these tests. We call these tests HAPTDUHs, mainly because the name sounded cool. In further work, we hope to extend our ideas to develop a revolutionary new class of games that better exploit the cycles of computers to solve problems that humans cannot yet solve. We call such games GWOPs (short for "Games With Out Purpose" [1]). These games have the potential to transform the parasitic relationship between computers and humans into a symbiotic oasis of corny references to The Matrix(TM).

References

[1] Luis von Ahn. **Games With a Purpose.** In IEEE Computer Magazine.

[2] Luis von Ahn, Manuel Blum, Nick Hopper and John Langford. **CAPTCHA: Using Hard AI Problems For Security.** In Eurocrypt 2003.

[3] Luis von Ahn, Manuel Blum and John Langford. **How Lazy Cryptographers do AI.** In Communications of the ACM, Feb 2004.