YAO'S MILLIONAIRES' PROBLEM AND PUBLIC-KEY ENCRYPTION WITHOUT COMPUTATIONAL ASSUMPTIONS

DIMA GRIGORIEV, LASZLO B. KISH, AND VLADIMIR SHPILRAIN

Abstract. We offer efficient and practical solutions of Yao's millionaires' problem without using any one-way functions. Some of the solutions involve physical principles, while others are purely mathematical. One of our solutions (based on physical principles) yields a public-key encryption protocol secure against (passive) computationally unbounded adversary. In that protocol, the legitimate parties are not assumed to be computationally unbounded.

1. Introduction

The "two millionaires problem" introduced by Yao in [6] is: Alice has a private number a and Bob has a private number b, and the goal of the two parties is to solve the inequality $a \leq b$? without revealing the actual values of a or b, or more stringently, without revealing any information about a or b other than $a \leq b$ or $a > b$. This latter requirement is somewhat informal because it appeals to an elusive concept of "information", so we will attempt to make it more formal in Section 6.

We note that all known solutions of this problem prior to 2014 (including Yao's original solution) used one-way functions one way or another. (Informally, a function is *one-way* if it is efficient to compute but computationally infeasible to invert on "most" inputs.) Therefore, these solutions are not applicable if Alice and Bob are computationally unbounded. An interesting question therefore is: does Yao's millionaires' problem have a solution if the two parties are computationally unbounded? In other words, is there a solution that is not based on any computational assumptions?

In [4], such solutions were offered based on various laws of physics. We recall one of them in Section 2 to put things in perspective, and in Section 3 we offer an equally simple new solution. We note that both these solutions, while otherwise quite practical, have one disadvantage: to implement either of them, Alice and Bob have to be (more or less) in the same place at the same time. In our Section 4 we offer a new solution, based on different laws of physics, that will allow the two parties to settle their dispute remotely.

Furthermore, and more importantly, the solution in Section 4 has other useful properties that allow us to extend the relevant protocol to an *informationally secure* public-key encryption protocol in Section 5. Specifically, if Alice transmits to Bob just one bit using our protocol in Section 5, then the probability for a passive adversary (even a computationally unbounded one) to determine what bit Alice intended to transmit, is exactly $\frac{1}{2}$, which means the protocol is perfectly secure against any passive adversary.

The fact that our solutions are "physical" should not make them look like a "curious but useless". They are, in fact, quite practical, although have a disadvantage of not being applicable if Alice and Bob are a long distance from each other.

That said, here we also offer two "purely mathematical" solutions of the millionaires' problem, without using any computational assumptions. These solutions are very efficient, too, because the parties do not really do any computation, and we argue that our solutions are, in fact, practical, i.e., they can be used in real-life situations, for example, in divorce settlement negotiations.

Research of Dima Grigoriev was supported by the RSF grant 16-11-10075. Research of Vladimir Shpilrain was partially supported by the NSF grant CNS-1117675 and by the ONR (Office of Naval Research) grant N000141512164.

In Section 7, we offer a solution where the probability for either party to guess the other party's integer correctly is $\frac{1}{\sqrt{n}}$, where $n = N_2 - N_1$. This probability is converging to 0 when n goes to ∞, although it converges somewhat slower than the "ideal" probability $\frac{\ln n}{n}$ does (see our Section 6). On the other hand, a nice property of this solution is that $\frac{1}{\sqrt{n}}$ is essentially the upper bound on the probability of a correct guess, so we have some kind of a *guarantee* of privacy (independent of *any* assumptions) in this case.

In Section 8, we offer another simple solution, based on a well-known method of *dichotomy*. Here we do not have a good upper bound on the probability of a correct guess, but on the other hand, the total probability for either party to guess the other party's number correctly is $\frac{\log_2 n}{n}$, which is only "slightly" higher (more precisely, $\log_2 e \approx 1.44$ times higher) than the "ideal" probability $\frac{\ln n}{n}$.

Both our "purely mathematical" solutions are very efficient and practical. The preference for either solution is determined by specific real-life applications. For example, if possible values of both a and b are uniformly distributed on the set of integers in $[N_1, N_2]$, then the "dichotomy" solution works better. In some other situations (e.g. in divorce settlement negotiations), where it is expected that possible values of a and b are reasonably close to each other, the preference goes to the solution in Section 7.

We also note that in his paper [6], Yao actually put forward a more general problem of *secure computation*, as follows. Suppose n people wish to compute the value of a function $f(x_1, \dots, x_n)$, which is an integer-valued function of n integer variables x_i of bounded range. Assume initially person P_i knows only the value of his x_i and no other x_j. Is it possible for them to compute the value of f, by communicating among themselves, without giving away any information about the values of their own variables? The millionaires' problem corresponds to the case where $n = 2$ and f is the sign function of $a - b$. The case where $n = 2$ is special because for $n = 2$, the general problem obviously does not have a solution for some functions $f(x_1, x_2)$, including $f(x_1, x_2) = x_1 + x_2$. Indeed, if, say, P_1 ends up knowing $x_1 + x_2$, then, since he knows his own x_1, he can recover $x_2 = (x_1 + x_2) - x_1$. The same happens for any function $f(x_1, x_2)$ such that $g(x_2) = f_{x_1}(x_2)$ is one-to-one for any fixed x_1. The function $sgn(x_1 - x_2)$, on the other hand, is not one-to-one for a fixed x_1, which is why the millionaires' problem makes sense. We note that for $n > 2$, many n-variable functions, including the sum, can be securely computed without computational assumptions, see e.g. [3].

1.1. **Adversary model.**

In our solutions of Yao's millionaires' problem, Alice and Bob are considered what is called "semi-honest" or "honest-but-curious", i.e., they can observe, measure, and compute whatever they like, but they respect all the protocol steps. A similar adversary model applies to our public-key encryption protocol in Section 5, see Section 5.1.

2. "Elevator" solution

This is logistically the simplest solution and it does not really use any laws of physics. Suppose there is an elevator building with at least $n = N_2 - N_1$ floors. Since Alice and Bob are computationally unbounded, they can build such a building if necessary.

Alice positions herself on the floor number a, and Bob gets to the floor number b. After that, Bob takes an elevator (Bob's private space) going down, stopping at every floor. Alice is just watching the elevator doors on her floor, making sure that Bob does not see her if the elevator doors open (here is Alice's private space). If she ever sees the elevator doors open, she knows that Bob's number is larger. If not, then his number is smaller.

We note that with this procedure, Bob will not know the result of comparison until Alice shares it with him. We also note that Alice may cheat by running between different floors to get a better estimate of Bob's number. If this is a concern, then a technical solution can be found. For example, Bob can lock the stairs and disable all elevators except one.

3. "Laboratory scale" solution

A laboratory scale is a simple mechanism with two plates that are in balance when no weight is placed on either of them.

Alice and Bob each manufacture a weight corresponding to their private number (in grammes or whatever units). Since they are computationally unbounded, they can do that whatever their numbers are. We also assume that they have identical boxes (their private space) where they can put their weight.

Now Alice enters the room where the scale is positioned and puts her box on one of the plates. Then Bob enters and puts his box on the other plate. If his plate goes down, then his number is larger; otherwise it is Alice's number that is larger.

We note that in this scenario, Alice and Bob do not have to be in the same place at the same time to perform the comparison, but they still have to be in the same place at some point, which may be inconvenient. In fact, if, say, Alice is worried about Bob cheating (by putting different weights on his plate to zoom in on Alice's weight), then she would have to stay in the room and watch what Bob is doing.

We also note that it may seem that placing a weight in a box is a "physical equivalent" of encryption by a one-way function. This, however, is not the case: all parameters of a box, including the weight, are known to the public, so the only purpose of the box is to protect the private weight from being observed. This is therefore more similar to hiding a private key in one's personal computer in a "standard" cryptographic protocol.

4. "Electrical circuit" solution

Here Alice and Bob each have, in their private space, a voltage generator U_A (respectively, U_B) and a resistor R_A (respectively, R_B). The resistance values are chosen randomly before the circuit is connected. When the circuit is connected, the electric current's absolute value is $|I| = \frac{|U_A - U_B|}{R_A + R_B}$ and its direction is toward the generator with the lower voltage.

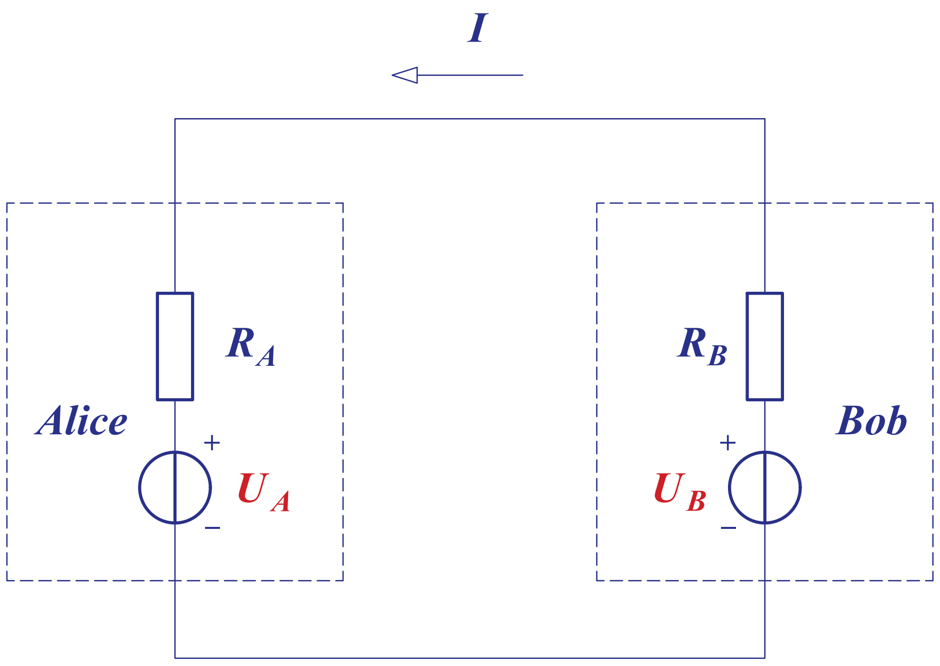

Figure 1. Electrical circuit

Thus, if Alice's private number is interpreted as U_A and Bob's private number is interpreted as U_B, then the direction of the electric current will determine whose number is larger.

From the security point of view, it may look like Alice has two equations in two unknowns U_B, R_B: one of them being the formula for I above, and the other one being Ohm's law that says that the difference of potentials between the upper and the lower "horizontal" wires is $U_B + IR_B$. However, the formula for I actually follows from Ohm's law: since the difference of potentials mentioned above is also equal to $U_A - IR_A$, we recover the formula for I from the equality $U_A - IR_A = U_B + IR_B$.

As far as a third party is concerned, she will have 3 equations with 4 unknowns, providing many solutions. In particular, these 3 equations will not give a third party any information about the value of U_A or U_B.

Note that security in this situation is not based on any computational hardness assumptions but instead is what we call "decoy-based" [4], i.e., we give to everybody the power to solve any computational problem, but the number of "decoy" solutions is large enough to make the probability of guessing the "real" solution negligible.

However, since we are going to convert this solution of Yao's problem to a public key encryption scheme in Section 5, we have to also address security of this solution (against a third party) during the *transient phase*, i.e., right after the circuit is connected and the voltage and current are changing. If the wire inductance L is public, then a third party can recover $R_A + R_B$ from the equation $\tau = \frac{L}{R_A+R_B}$, where τ is the relaxation time constant. (In general, also cable capacitance and propagation time delay can produce similar effects.) To prevent this from happening, Alice and Bob should randomly oscillate their voltage and fluctuate resistance during the transient phase, and let the other party know when they stop doing that. Note that the range and the statistical and dynamical properties of these independent fluctuations must be chosen reasonably (i.e., they are not completely arbitrary) to minimize information leak. In particular, this should not be just a monotone increase, but rather a combination of an intermittent increase with an oscillation of dynamical features related to the relaxation time mentioned above.

Figure 2 below shows the relevant electrical circuit.

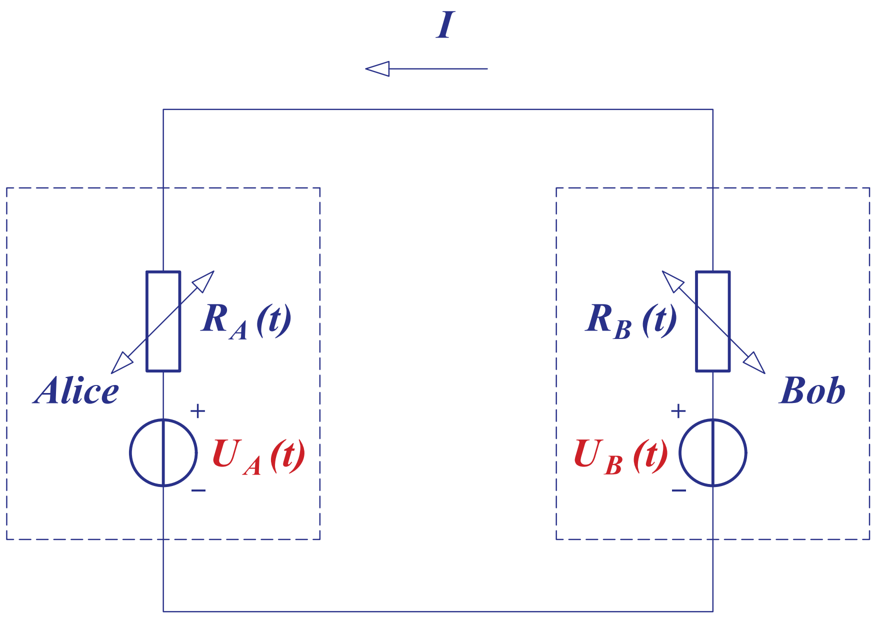

FIGURE 2. Fluctuating voltage and resistance during the transient phase

5. Public key encryption from a solution of the millionaires' problem

We now describe a way to obtain a public key encryption protocol from a solution of the millionaires' problem given in our Section 4. What makes this possible is:
(1) A third party (an adversary) cannot determine Alice's or Bob's private number by observing the public space. This is not the case, for example, with the "elevator" solution because by observing the public space, the adversary will see where Bob started his descent.
(2) The adversary ends up only with the knowledge that $a > b$ or $a < b$, but this does not narrow down the range for either a or b for the adversary. On the other hand, Alice, for example, having learned that $a < b$, is able to narrow down the range for b to $[a, N_2]$. This advantage is lacking in "purely mathematical" solutions in our Sections 7 and 8. There an adversary observing the protocol will end up with the same knowledge about the range for private numbers as Alice or Bob will.

To make it simpler, we assume in the protocol below that Alice wants to send just a single bit c to Bob. The probability for the adversary (even a computationally unbounded one) to determine what bit Alice intended to transmit, is going to be $\frac{1}{2}$, which means the protocol is perfectly secure.

(1) **Key generation.** Bob randomly selects a private integer b in the public range $[N_1, N_2]$. This b is his decryption key.
(2) Alice randomly selects a number a in the range $[N_1, N_2]$. To avoid a possible "collision" $a = b$, Alice selects her a in the form $(k + \frac{1}{2})$, where k is an integer.
(3) Alice and Bob run the protocol from Section 4 to determine whether $a > b$ or not. Suppose $a > b$.
(4) Alice randomly selects a number a_1 in the range $[N_1, a]$. Bob keeps the same b.
(5) Alice and Bob run the protocol from Section 4 to determine whether $a_1 > b$ or not. If $a_1 > b$, they repeat from Step 4. If $a_1 < b$, then Alice knows that Bob's b is in the range $[a_1, a]$. If, however, the length of $[a_1, a]$ is more than $\frac{1}{3}$ of the length of $[N_1, N_2]$, then Alice aborts the protocol and the parties start over.
(6) Let S be the set of all integers in the interval $[a_1, a]$. In the complement of $[a_1, a]$ in the interval $[N_1, N_2]$, Alice randomly selects another interval $[x, y]$ of the same length as $[a_1, a]$. Let S' be the set of all integers in the interval $[x, y]$.
(7) To send a bit c to Bob, Alice makes up a polynomial $P(x)$ (using Lagrange interpolation, say) that takes value c on every integer from the set S and value $1 - c$ on every integer from the set S'. This $P(x)$ is therefore the *ciphertext Alice then sends the polynomial* $P(x)$ *to Bob.*
(8) *Bob computes* $P(b) = c$.

Note that the interval $[x, y]$ here plays the role of a "decoy". A computationally unbounded adversary can plug all integers from $[N_1, N_2]$, one at a time, in the polynomial $P(x)$. He will then see that on integers from the set S, $P(x)$ is equal to c, while on integers from the set S', $P(x)$ is equal to $1 - c$. Since there is no reason for the adversary to prefer one interval over the other, all he can do is take a guess that will be successful with probability $\frac{1}{2}$.

5.1. Adversary model.

We emphasize that in this paper, we only address security against a *passive* adversary, i.e., an adversary who can observe, measure, and compute, but cannot interfere with the protocol execution. Addressing attacks by an active adversary (e.g. the "man-in-the-middle" attack), as well as countermeasures against such attacks would take us too far into the world of physics, while in this paper, we would like to focus on what we believe is a conceptually new idea of information-theoretic security (i.e., security without computational assumptions) of public-key encryption.

5.2. **Security.** We emphasize once again that security of the above protocol is what we call "decoy-based", i.e., from the adversary's perspective there are several (in our case here, just two) possibilities for the decryption key, and therefore also for the result of decryption, with no reason to prefer the actual one over the others ("decoys"). We note that this kind of security was used before in weaker cryptographic primitives, namely in secure delegation of computation, see e.g. [1], [2], [5], although the name "decoy-based" was not used there.

However, in a strong cryptographic primitive such as public-key encryption, "decoy-based" security was not widely considered, see discussion in [4]. Moreover, in all existing proposals (some of them only relatively recently de-classified) laws of (classical) physics were employed, which makes "standard" security analysis essentially inapplicable. Moreover, in all these proposals there is no public encryption key in the usual sense, although there is a decryption key; in our protocol, the decryption key is b, Bob's private number. Thus, strictly speaking, "public-key encryption" is not an accurate description of these protocols; a better wording would be "asymmetric encryption". It is just that "public-key encryption" has become almost a household name for whatever encryption that is not symmetric, which is why we use this wording here, by somewhat abusing terminology.

Back to the particular encryption protocol in this section, its security is clearly not based on any computational assumptions, but rather on an assumption from physics. Namely, the assumption is that there are no other laws of physics (other than Ohm's law) that can provide the adversary with additional information about the parties' private numbers. If one accepts this assumption, the rest is quite straightforward: the adversary just has to guess, with equal probability, between two given subintervals, where one of the subintervals is "real", while the other one is a "decoy". This implies guessing the transmitted bit with probability $\frac{1}{2}$.

5.3. **Efficiency.** While we claim security against (passive) computationally unbounded adversary, legitimate parties (Alice and Bob) do not have to be computationally unbounded to execute the protocol in this section. A curious reader may therefore wonder how performance of our protocol compares to that of well-established public-key encryption protocols. To that end, we first mention that the length of the interval $[N_1, N_2]$ can be 100 (or even less), in which case the degree of the polynomial $P(x)$ will be under 50. For convenience, let us assume that the degree of $P(x)$ is exactly 50. If Alice uses Lagrange interpolation to build $P(x)$ based on values at integer points from a subinterval of [1, 100], then coefficients of $P(x)$ are going to be on the order of 2^{100} (on average), and therefore the size of the whole $P(x)$ (recall that $P(x)$ is the ciphertext) is on the order of 5000 bits. This is comparable to the size of the ciphertext in the RSA, say. Also, evaluating all the coefficients of $P(x)$ takes the total of about 1500 multiplications of integers not exceeding 100, which is more efficient than computation involved in modern implementations of the RSA is. However, in the RSA a ciphertext of size 5000 bits can be encryption of a plaintext of the same size, whereas in our scheme, we use it to encrypt justx a single bit. This is the price to pay for security against computationally unbounded adversary. From practical point of view, it is therefore more reasonable to use our scheme for a shared secret key establishment rather than directly for encryption.

6. A possible (probabilistic) model for the millionaires' problem

In this and the following sections, we discuss "purely mathematical" solutions of Yao's millionaires' problem, without any computational assumptions.

First of all, we note that, say, Bob might learn Alice's private integer a even if he did not intend to. For example, if the range for a and b is $[N_1, N_2]$, and Bob's integer b happens to be equal to N_1, then, after having found out that $a \leq b$, Bob knows that Alice's integer is $a = N_1$. Then, if $b = N_1 + 1$, the information $a \leq b$ tells Bob that either $a = N_1$ or $a = N_1 + 1$, so he can guess a correctly with probability 1/2. Thus, assuming (just for simplicity of computation) that *a priori* both a and b are random variables uniformly distributed on the set of integers in $[N_1, N_2]$, in the "ideal" situation where an oracle just tells Bob that, say, $a \le b$, the total probability for Bob to guess a correctly is:

$$\frac{1}{n}\sum_{k=1}^{n}\frac{1}{k},$$

where $n = N_2 - N_1$.

This sum is asymptotically equal to $\frac{\ln n}{n}$, and this therefore should be considered the "ideal" solution in this probabilistic model. If probability distributions of a and b are unknown, then the *a priory* probability of guessing is the same as it is in the case of uniform distribution.

Thus, in the absence of an oracle, what one can hope for is:

> Design an information exchange protocol between Alice and Bob so that after this protocol is executed, Alice and Bob know whether or not $a \le b$, but the probability for either party to guess the other party's integer correctly converges to 0 when n goes to infinity, where $n = N_2 - N_1$.

7. First mathematical solution

Here we offer a very efficient and simple, almost naive, solution that achieves the goal described in Section 6, but is not quite satisfactory from the practical point of view, as we explain at the end of this section. In the following Section 8, we are able to improve this solution to make it of practical value.

Here is the protocol.

(1) Bob begins by breaking the set of n integers from the interval $[N_1, N_2]$ into approximately $\sqrt{n}$ subintervals with approximately $\sqrt{n}$ integers in each, in such a way that his integer b is an endpoint of one of the subintervals.
(2) Bob then sends the endpoints of all the subintervals to Alice.
(3) Alice tells Bob in which subinterval her integer a is. By the above property of Bob's subintervals, all elements of the subinterval pointed at by Alice are either less than (or equal to) b or greater than b, so Bob now has a solution of the inequality $a \le b?$, and he can share it with Alice.

It is obvious that the probability for Bob to guess Alice's integer a correctly, as well as the probability for Alice to guess Bob's integer b correctly, is approximately $\frac{1}{\sqrt{n}}$. This probability is converging to 0 when n goes to ∞, although it converges somewhat slower than $\frac{\ln n}{n}$ does.

We also note that this protocol is very efficient. The parties do not perform any real computation, and the amount of transmitted data is quite small. Indeed, Bob can transmit to Alice just, say, the right endpoint of the leftmost subinterval and the length of all other subintervals, assuming he makes all of them have the same length. Alice transmits just two endpoints of "her" subinterval.

There is one problem with this solution, however. It is the fact that Alice narrows down the range for her number a too much when she tells Bob in which subinterval her a is. By contrast, possible values of Bob's number b are "well spread" over the whole interval $[N_1, N_2]$. Thus, intuitively (think divorce settlement negotiations) Bob is in a better position here. For example, if the range $n = N_2 - N_1$ is \$1M, then what Alice ends up knowing is just that Bob's wealth is represented by a whole number of thousands of dollars, which is not a very useful information (in the context of settlement negotiations, say). Bob, on the other hand, ends up knowing that Alice's wealth is represented by a number between k and $k + 1$ thousand dollars, which is almost as good as precise information about Alice's wealth.

Thus, the lesson here is: the probabilistic model in our Section 6 is not quite satisfactory from the practical point of view because it is not just the probability of guessing the opponent's number that might matter, but also the "spread". In the next Section 8, we offer another solution of the millionaires' problem, where the *a priory* (i.e., before execution of the protocol) probability for either party to guess the opponent's number correctly is asymptotically $\frac{\log_2 n}{n}$ and the "spread" of possible values (after execution of the protocol) is the same for both parties.

8. Second mathematical solution

We now give a solution of the millionaires' problem, which is quite different from the solution in Section 7. The method we use is a well-known *dichotomy.*

(1) Alice tells Bob in which half of the interval $[N_1, N_2]$ her number a is. If Bob's number b is in the other half (this happens with probability $\frac{1}{2}$), then the problem is solved, and the probability for either party to guess the other party's number correctly is $\frac{2}{n}$.
(2) If Bob's number is in the same half of the interval $[N_1, N_2]$, then Alice tells Bob in which half of this half-interval her number is. Again, if Bob's number is in the other half (this happens with probability $\frac{1}{4}$), then the problem is solved, and the probability for either party to guess the other party's number correctly is $\frac{4}{n}$.
(3) Alice and Bob continue with this dichotomy until either their numbers happen to be in different subintervals or turn out to be equal.

Thus, if the protocol terminates after k steps, the probability for either party to guess the other party's number correctly is $\frac{2^k}{n}$. Now the question is: what is the expected value of the number of steps in this protocol, assuming that both a and b are random variables uniformly distributed on the set of integers in $[N_1, N_2]$? Since the probability of terminating after exactly k steps is $\min\{\frac{1}{2^k}, \frac{1}{n}\}$, the answer is $\sum_{k=1}^{\log_2 n} \frac{k}{2^k}$, which is asymptotically (when n goes to infinity) equal to 2, i.e., the protocol will most likely terminate in just 2 steps, with both parties knowing an interval of length $\frac{n}{4}$ where the opponent's number should be.

We also note that the *average* length of an interval where either party can narrow down the other party's number location is $\sum_{k=1}^{\log_2 n} \frac{1}{2^k} \cdot \frac{n}{2^k}$, which is asymptotically equal to $\frac{n}{3}$, with a satisfactory "spread" over a subinterval of length $\frac{n}{3}$ for values of either party's number. Of course, a disadvantage of this solution is that, if two private numbers are rather close to each other and the dichotomy protocol halts after k steps, then the probability $\frac{2^k}{n}$ for either party to guess the other party's number correctly *after* execution of the protocol can be rather close to 1. However, in the scenario where both numbers are assumed to be uniformly distributed over the whole range, we have the following fact, which is probably well-known:

Fact. If a and b are independent random variables and each is uniformly distributed on $\{1, 2, \ldots, n\}$, then the expected value of $|a-b|$ is $E(|a-b|) = \frac{(n^2-1)}{3n}$, which is asymptotically equal to $\frac{n}{3}$.

Indeed, note that $|a-b| = \max(a,b) - \min(a,b)$. By symmetry, $E(\max(a,b)) = n+1-E(\min(a,b))$, hence $E(|a-b|) = n+1-2E(\min(a,b))$. Now direct computation gives $E(\min(a,b)) = \sum_{k=1}^{n} k \cdot (\frac{2}{n} \cdot \frac{n-k}{n} + \frac{1}{n^2}) = n+1+\frac{(n+1)(1-4n)}{6n}$. Then $E(|a-b|) = n+1-2E(\min(a,b)) = \frac{(n^2-1)}{3n}$.

Thus, a and b are likely to be sufficiently far apart, which explains why the above protocol terminates after just 2 steps on average.

We also note that the *a priori* total probability for either party to guess the other party's number correctly in this scenario is given by the sum $\sum_{k=1}^{\log_2 n} \frac{1}{2^k} \cdot \frac{2^k}{n} = \sum_{k=1}^{\log_2 n} \frac{1}{n} = \frac{\log_2 n}{n}$, which is only "slightly" higher (more precisely, $\log_2 e \approx 1.44$ times higher) than the "ideal" probability $\frac{\ln n}{n}$, see our Section 6.

Acknowledgement. D. Grigoriev and V. Shpilrain are grateful to Max Planck Institut für Mathematik, Bonn for its hospitality during the work on this paper. D. Grigoriev is also grateful to MCCME for wonderful working conditions and inspiring atmosphere.

References

[1] B. Cavallo, G. Di Crescenzo, D. Kahrobaei, V. Shpilrain, *Efficient and secure delegation of group exponentiation to a single server*, in: RFIDsec 2015, Lecture Notes Comp. Sc. **9440** (2015), 156–173.

[2] M. Dijk, D. Clarke, B. Gassend, G. Suh, and S. Devadas, *Speeding Up Exponentiation using an Untrusted Computational Resource.* In: Designs, Codes and Cryptography, 39 (2006), 253–273.

[3] D. Grigoriev and V. Shpilrain, *Secrecy without one-way functions*, Groups, Complexity, and Cryptology **5** (2013), 31–52.

[4] D. Grigoriev and V. Shpilrain, *Yao's millionaires' problem and decoy-based public key encryption by classical physics*, Int. J. Foundations Comp. Sci. **25** (2014), 409–417.

[5] S. Hohenberger and A. Lysyanskaya, *How to securely outsource cryptographic computations,* in: Theory of Cryptography (TCC 2005), Lecture Notes Comp. Sc. **3378** (2005), 264–282.

[6] A. C. Yao, *Protocols for secure computations* (Extended Abstract), 23rd annual symposium on foundations of computer science (Chicago, Ill., 1982), 160–164, IEEE, New York, 1982.

CNRS, Mathématiques, Université de Lille, 59655, Villeneuve d'Ascq, France
E-mail address: `dmitry.grigoryev@math.univ-lille1.fr`

Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843
E-mail address: `laszlokish@tamu.edu`

Department of Mathematics, The City College of New York, New York, NY 10031
E-mail address: `shpil@groups.sci.ccny.cuny.edu`