Genetic Algorithm to Study Practical Quantum Adversaries

Walter O. Krawec
University of Connecticut
Storrs, CT
walter.krawec@uconn.edu

Sam A. Markelon
University of Connecticut
Storrs, CT

ABSTRACT

In this paper we show how genetic algorithms can be effectively applied to study the security of arbitrary quantum key distribution (QKD) protocols when faced with adversaries limited to current-day technology. We compare two approaches, both of which take into account practical limitations on the quantum power of an adversary (which can be specified by the user). Our system can be used to determine upper-bounds on noise tolerances of novel QKD protocols in this scenario, thus making it a useful tool for researchers. We compare our algorithm's results with current known numerical results, and also evaluate it on newer, more complex, protocols where no results are currently known.

CCS CONCEPTS

• **Security and privacy** → **Cryptography**; • **Computing methodologies** → **Search methodologies**;

KEYWORDS

Genetic Algorithm, Quantum Computing, Quantum Cryptography

ACM Reference Format:
Walter O. Krawec and Sam A. Markelon. 2018. Genetic Algorithm to Study Practical Quantum Adversaries. In *GECCO '18: Genetic and Evolutionary Computation Conference, July 15–19, 2018, Kyoto, Japan,* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.1145/3205455.3205478

1 INTRODUCTION

Quantum Key Distribution (QKD) protocols allow for the establishment of a secret key between two users (Alice A and Bob B) which is secure against all-powerful adversaries. This task is impossible using classical communication only; indeed, our entire modern-day communication infrastructure depends entirely on certain computational assumptions remaining true. However, by using quantum communication, it is possible to have security assuming only that the adversary obeys the laws of physics (a much safer assumption than the difficulty of factoring a number into two primes). Furthermore, QKD is a practical, real-world, technology today with several companies producing commercial quantum equipment. For more information, the reader is referred to [11].

Currently, numerous QKD protocols exist, many with unconditional proofs of security. Security proofs against all-powerful adversaries generally involve highly complex mathematical arguments. Many newer protocols, utilizing high-dimensional quantum carriers, or multi-pass quantum channels are conjectured to have several practical benefits (either by being cheaper to produce, having higher efficiency, or holding interesting security properties which may be useful for other cryptographic primitives); however, due to their complex nature, a full security proof is lacking. A security proof of a QKD protocol involves an information theoretic argument to compute its *key-rate* which is, roughly, the ratio of secure secret key bits to quantum bits sent. This key-rate computation should be a function only of observed noise in the quantum channel. The fascinating property of quantum communication, as opposed to classical communication, is that, the more noise in the quantum channel, the more information an adversary potentially has (and, so, the smaller the key-rate will be). Thus, one wishes to bound the adversary's information as a function of the noise in the quantum channel. Eventually, if the channel is too noisy, the key-rate becomes zero - this threshold noise value is called a QKD protocol's *noise tolerance*.

Evolutionary algorithms have been used for some time to evolve quantum algorithms [1, 10, 12–14] and to study classical cryptography [9]. In [6], a genetic algorithm was used to verify correctness and determine upper-bounds on security of arbitrary QKD protocols *against all-powerful adversaries*. This algorithm evolved attack operators modeled as unitary operators acting on a $T \cdot M$ dimensional complex vector space where T is the dimension of the quantum carrier used (e.g., dimension 2 if *qubits* are used [8]) and M is the dimension of the adversary Eve (E's) *quantum memory*. This approach was able to find the correct noise tolerance for protocols where theoretical values are known; it was also able to find noise tolerances for protocols which had no proof of security.

However, the work in [6] assumed the adversary had access to a perfect quantum memory, of exponential size, and was able to perform a theoretically optimal measurement of the entire memory at once (this measurement strategy was not part of the GA and its existence was simply assumed). Such technology is not currently available, and will likely not be available for many decades. Thus, it makes sense to also

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5618-3/18/07…$15.00
https://doi.org/10.1145/3205455.3205478

analyze protocols against *practical* adversaries which the algorithm in [6] was not designed to handle.

There are several ways to model such practical adversaries. Here, we focus on a model introduced in [2] which was called a memory-less eavesdropper, whereby E had access only to a very short-term quantum memory before she must make a measurement on it (the memory only lasted as long as the time it took for the qubit to pass through her lab). To analyze a protocol, using this approach, the authors first converted it to an entanglement based version and subsequently used semi-definite programming to determine optimal attack operations. This approach produced highly accurate noise tolerances for the protocols considered in that paper; however this approach is complex to use, and can only be used for protocols which admit an equivalent entanglement-based version (for many newer protocols relying on multi-pass channels, it is not currently known how to convert to an entanglement based version).

We make several contributions in this work. First, we show how a gate-based solution representation can be used to study practical quantum adversaries against *arbitrary* QKD protocols. *Our approach can be suitably modified to analyze practical adversaries beyond the memory-less eavesdropper model.* We compare this with a unitary based solution representation. We show that evolutionary methods can produce the same or similar noise tolerances as current-known results, thus giving evidence that our approach of using evolutionary computation techniques is correct. We also show our approach can be used for protocols which do not admit a known entanglement based version. Finally, our approach does not require extensive technical knowledge of the mathematical foundations of quantum computation and communication (necessary to use other numerical systems such as the one in [2]) making it potentially more applicable to a wider user base. As such, this approach may also be of interest to practitioners of quantum communication equipment, as analyzing protocols against practical adversaries can increase the efficiency of the communication rate (since an attacker has less information than if she had a perfect quantum memory).

2 QUANTUM COMMUNICATION AND KEY DISTRIBUTION

We now provide some general information on quantum communication, information, and key distribution. Due to length constraints, this section is necessarily short; however the reader is referred to [8, 11] for more information.

Classical bits can exist in one of two states: 0 and 1. Quantum bits (or *qubits*), however, may be prepared in infinitely many states of the form $\alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ subject, usually, to the normalization constraint $|\alpha|^2 + |\beta|^2 = 1$. We call the $|\cdot\rangle$ a *ket*.

Mathematically, an arbitrary *pure* quantum system is an element $|\psi\rangle$ in a Hilbert space. Since all systems of interest to us here are finite dimensional, we may equivalently consider $|\psi\rangle$ to live in the vector space $\mathbb{C}^n$. If $\{|0\rangle, \cdots, |n-1\rangle\}$ is an orthonormal basis of $\mathbb{C}^n$ (typically called the *computational basis* or Z basis), then an arbitrary quantum system may be represented as a linear combination (called a *superposition*) of these basis states; that is $|\psi\rangle = \sum_i \alpha_i |i\rangle$, subject to $\sum_i |\alpha_i|^2 = 1$. As vectors, one may consider the computational basis to be the standard Euclidean basis; however the choice is irrelevant.

Unlike a classical bit, which may be read at any time to determine, with perfect certainty, its state (0 or 1), "reading" a quantum state (called *measuring*) is a probabilistic process. Given $|\psi\rangle$ as in the previous paragraph, performing a measurement in the computational basis yields outcome $|i\rangle$ with probability $|\alpha_i|^2$. Furthermore the state collapses to the observed basis vector. Note also that qubits cannot be copied without potentially disturbing their state (unlike classical bits).

Often, we wish to consider two (or more) quantum systems. If $|\psi_1\rangle \in \mathcal{H}_1 \cong \mathbb{C}^n$ and $|\psi_2\rangle \in \mathcal{H}_2 \cong \mathbb{C}^m$, then we model the joint state as $|\psi_1\rangle \otimes |\psi_2\rangle$ (which we often denote simply $|\psi_1\rangle|\psi_2\rangle$ or $|\psi_1, \psi_2\rangle$). This lives in the Hilbert space $\mathcal{H}_1 \otimes \mathcal{H}_2 \cong \mathbb{C}^{n \cdot m}$. In general, if $|\psi_1\rangle = (\alpha_1, \cdots, \alpha_n)^T$ (transposed as we typically view kets as column vectors), then $|\psi_1\rangle \otimes |\psi_2\rangle = (\alpha_1 |\psi_2\rangle, \cdots, \alpha_n |\psi_2\rangle)^T$.

While measurements cause a system to irreversibly collapse to the observed state, another operation allowed by the laws of quantum physics is *state evolution* via a unitary operator. An operator U is unitary if $UU^* = U^*U = I$ (where U^* is the conjugate transpose of U). The state $|\psi\rangle$, after evolution via U, is denoted $U|\psi\rangle$ and, mathematically, is simply the matrix product of U with the vector $|\psi\rangle$. Note that this evolution is reversible (the inverse of U is simply U^* which is also unitary).

A *quantum circuit* is a collection of *gates*, each gate a unitary operator typically acting on only one or two qubits. Typical gates include the Hadamard gate, the CNOT gate, and the rotation gate which will be defined later when used by our algorithm. Quantum circuits act on several "wires" each wire holding a single qubit. Thus, a circuit acting on m wires is also a unitary operator of dimension $2^m \times 2^m$.

For every "ket" $|\psi\rangle$ there is a corresponding "bra" $\langle\psi|$ which, in the finite dimensional case, is simply the conjugate transpose of the ket (i.e., $\langle\psi| = (|\psi\rangle)^*$) the terminology is Dirac's braket notation - a play on the word bracket. We denote by $\langle\psi|\phi\rangle$ to be the inner product of vectors $|\psi\rangle$ and $|\phi\rangle$. We denote by $|\psi\rangle\langle\phi|$ to be the outer product. Note that in both cases, the operation is simply matrix multiplication with the former yielding a 1×1 matrix (which can be taken to be a scalar in $\mathbb{C}$) and the latter an $n \times n$ matrix.

Every closed quantum system may be represented as a vector $|\psi\rangle$ in some Hilbert space. However, we often wish to model more general systems such as statistical ensembles of pure states. For instance, in our protocols, we will have parties sending pure qubit states $|\psi_i\rangle$ with certain probabilities p_i. Mathematically, we represent such a system using the density operator formalism. The above system is represented by the matrix $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$. More generally, a *density operator* is a positive semi-definite operator of unit trace; any quantum

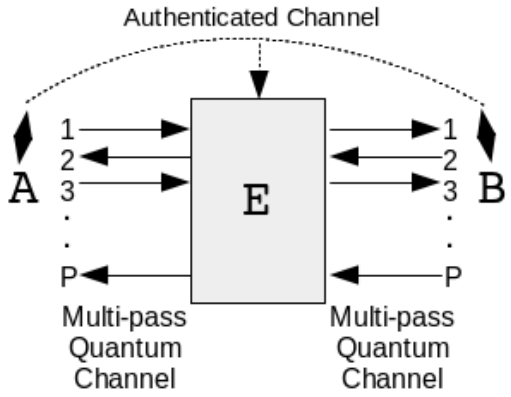

Figure 1: A typical QKD protocol utilizing a multi-pass channel.

system may be modeled as a density operator (including a pure state $|\psi\rangle$ which is the density operator $|\psi\rangle\langle\psi|$).

If ρ_{AE} is a density operator acting on Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_E$ (i.e., ρ_{AE} models a quantum system held by two parties A and E), then we write ρ_E to mean the result of tracing out A's portion; i.e., $\rho_E = tr_A \rho_{AE}$ and it describes the state of the system that only E holds. If $\rho_{AE} = \sum_{i,j} |i\rangle\langle j|_A \otimes \rho_E^{(i,j)}$ then $\rho_E = \sum_i \rho_E^{(i,i)}$.

Let $\rho = \rho_{W_1 W_2 \cdots W_m}$ be a quantum system on m wires, each wire holding a qubit (each wire is two-dimensional - thus the entire system is of dimension 2^m). Note that, if all m wires of ρ are measured, the system is no longer quantum, but is classical. If party A "holds" wires $\{w_i\}_{i\in\mathcal{A}}$ and party E holds wires $\{w_i\}_{i\in\mathcal{E}}$ (where by "hold" we mean the information contained in the wire, which is now a classical bit, is known to the respective party) then we write $H(A|E)_\rho$ to be the Shannon entropy of the classical system ρ_A conditioned on the classical system ρ_E (post measurement, these are simply classical random variables). If the context is clear, we will not write the subscript "ρ."

2.1 Quantum Key Distribution

QKD protocols consist of a quantum communication stage and a classical post-processing stage. The first takes place over numerous iterations (each iteration being treated identically; random choices are allowed, but the distribution of the choices are fixed for each iteration). This stage utilizes a *multi-pass quantum channel*, allowing qubits to travel from A to B (or vice versa) multiple times each iteration (generally, this is a fixed, protocol specific value we denote P in which case we say the channel is a P-pass quantum channel). This stage also uses an authenticated classical channel which allows A and B to read and write classical messages to one another; however E may only listen and not send messages of her own (thus, it is not a secret channel). The goal of this stage is to output a *raw-key* of size N bits and also an estimate of the noise (e.g., error rate) in the quantum channel. See Figure 1.

The raw-key that is output is partially correlated between A and B and partially secret. Thus, the second stage, classical post-processing, will first run an error correcting protocol (over the authenticated classical channel, thus leaking additional information to E). Following this, a privacy amplification protocol is run, taking the error-corrected raw-key and distilling a secret key of size $\ell(N) \leq N$ (possibly $\ell(N) = 0$ if E has too much information). These are standard processes and for more information, the reader is referred to [11]. We are interested in determining the ratio $r = \frac{\ell(N)}{N}$ as $N \to \infty$. This r is called the *key-rate* in the *asymptotic scenario*.

In the case of memory-less adversaries, E will attack each iteration of the quantum communication stage independently and identically by "probing" the quantum channel using a quantum circuit (or general quantum unitary operator). This probe acts on the qubit (traveling on what we call the *Transit Wire*), a special "guess" wire, and a short-term quantum memory that will be discarded as soon as the iteration is complete. In the case of multi-pass channels, E will apply a (possibly different) probe on subsequent passes, acting on the same wires. Finally, the adversary will discard her auxiliary wires and measure the "guess" wire which will be her guess as to the raw key-bit that A is trying to send to B. Below, we use E to denote the random variable of this guess wire (which can take only two values 0 or 1). E's goal is to find an attack which induces minimal noise, that maximizes the chance she "guesses" correctly (i.e., $A = E$). In essence, she wants to learn as much as she can without being "caught".

Since her attack is the same each iteration (we comment later on how this assumption may be removed by our system), and since she is forced to measure her system at the end thus collapsing a quantum system to a classical one (an assumption made to model a practical adversary), we may use results from Csiszar-Korner [4] to evaluate r, namely:

$$r = \lim_{N\to\infty} \frac{\ell(N)}{N} = H(A|E) - H(A|B). \quad (1)$$

So long as $r > 0$, a secret key may be distilled after privacy amplification. Assuming the protocol is secure, as the noise increases, r inevitably decreases (as E's uncertainty, measured by $H(A|E)$, also should decrease assuming she is using an optimal attack). The noise level at which r becomes zero is called the protocol's *noise tolerance*. If a protocol is insecure, then $r = 0$ with no noise induced (i.e., its noise tolerance is 0%). We will apply evolutionary algorithms to search for optimized practical attacks causing minimal noise, but maximizing E's probability of guessing A's key bit.

3 THE ALGORITHM

Our goal is to devise a system whereby users may input an arbitrary QKD protocol and have as output an estimate of this protocol's noise tolerance when faced with memory-less attackers. To do so, we construct a GA which evolves actual attack strategies (which consist of a quantum probe and a measurement strategy) attempting to find one which both minimizes the induced noise, while maximizing E's information on A and B's raw-key. In particular, the attack

evolved must consist of an output "guess" as to what the key-bit is. This is in contrast to previous work [6] where general, arbitrary, unitary operators, representing E's attack, were evolved and it was assumed that E performed a theoretically optimal measurement to extract maximal information from the resulting exponential sized quantum memory (which may not be practical to implement even with future technology).

3.1 Solution Representations

In general, a candidate solution is a potential attack against the QKD protocol under analysis. We test and compare two solution representations in this work. One based on evolving quantum circuits out of primitive gates; the second evolving arbitrary quantum operators. Both must incorporate a strategy so that, once measured, E's guess wire is highly correlated with A's key-bit (despite having lost access to any auxiliary wires used), while inducing minimal noise. The circuit based approach gives us high flexibility in studying practical adversaries; the unitary based approach is the model considered in [2] and also gives E potentially more power (if the circuit size is small for the first).

First Solution Representation - The first solution representation we consider is based on one introduced in [10] originally meant to evolve optimized quantum algorithms. In our case, it will be used to represent a quantum attack strategy consisting of multiple quantum gates and measurement operations. In more detail, E has access to $M+1$ wires where M is a constant specified by the user. The larger M is, the more (temporary) quantum processing power she has. The additional "+1" wire is used to output E's guess as to whether A's key-bit is a 0 or a 1 for the iteration being attacked. In this work, we consider only attack strategies whereby E performs the same operation each iteration of the protocol (these may be randomized attacks - however the distribution she uses is identical each iteration). Our approach can be extended to handle the case where E attacks blocks of $B > 1$ sequential iterations differently (i.e., generating a sequence of B different attacks and recycling them every B iterations). The method we describe would apply simply by creating B "guess" wires and using $M + B$ wires. However, due to the exponential increase in memory required to simulate a quantum system, we did not attempt to perform experiments involving this case, leaving this analysis for future work. Note that, in QKD literature, it is very common to consider independent, identical attacks (i.e., $B = 1$) when analyzing "practical" attacks [2]. Keep in mind that as B increases, the quantum power of the adversary also increases beyond what is considered currently practical.

Let $\mathcal{G}$ be a set of allowed quantum gates. For our evaluations, we used four gates - three quantum and one measurement. The quantum gates we use are the Hadamard $\hat{H}$, a general unitary gate R, and a $CNOT_{X,Y}$ gate which flips the Y wire if the X wire is a $|1\rangle$ (otherwise it leaves it untouched). As matrices, these are:

$$\hat{H} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{2}$$

$$CNOT_{X,Y} = |0\rangle\langle 0|_X \otimes I_Y + |1\rangle\langle 1|_X \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}_Y \tag{3}$$

$$R(p,\theta,\psi) = \begin{pmatrix} \sqrt{p}e^{i\theta} & -\sqrt{1-p}e^{-i\psi} \\ \sqrt{1-p}e^{i\psi} & \sqrt{p}e^{-i\theta} \end{pmatrix} \tag{4}$$

While we chose these gates, others may be easily added (or removed) arbitrarily. Note that if an additional so-called $\pi/8$ gate is added, the set becomes *universal* (that is, any quantum operation U may be constructed, to arbitrary precision, from a suitably sized collection of gates in this set; the amount of gates required in a circuit to approximate U depends on the desired precision). Though we did not test this in our evaluations below, it is easy to add this additional gate. However, without this gate, we got good results in our evaluations. We also include a Z basis measurement operation (if E wishes to measure in an alternative basis, this can be done very easily by first applying R and then a measurement). Thus $\mathcal{G} = \{\hat{H}, CNOT, R, Z\}$ where Z denotes the Z basis measurement operation. Note that, if new gates are added, it may make the GA's task of finding a solution easier (though at the cost of increasing the complexity of E's "practical" attack). To add a gate, one must simply add its functionality to our simulator (discussed next) and the GA will automatically incorporate the new gate in its evolved solutions. Note that entire quantum algorithms can be included as a "gate" in $\mathcal{G}$.

To represent a gate, we require its type (in our implementation, the type is simply an integer between 0 and $|\mathcal{G}| - 1$). We also require at most two indices representing which wire it should be applied to (the CNOT gate requires two wires: X and Y; the others require only one). This is an index between 0 and $M + 1$ (inclusive; thus there are $M + 2$ wires in total). Index 0 is assumed to be the transit wire (connecting A and B) for the pass being attacked; index 1 is E's "guess" wire; the additional M wires are E's auxiliary space which she may use arbitrarily in her attack (though they will be measured and discarded later leaving only her "guess" as shown in Figure 2). Finally, several "attributes" are also potentially needed which are stored as double precision floating point values (these are used, for example, to store the values of p, ψ, and θ in the R gate).

A list of these gates $(G_1, G_2, \cdots, G_K)$ represents an attack strategy on one pass (the strategy consists simply of E applying gate G_1 followed by G_2 and so on). Of course, we are also interested in analyzing multi-pass QKD protocols (allowing E to alter her attack on pass i based on the attack results from pass $j < i$). Thus, a candidate solution is an array (of size P, where P is the number of passes a QKD protocol makes each iteration) of lists of gates $\{(G_1^j, \cdots, G_{K_j}^j)\}_{j=1}^P$ with probe $(G_1^j, \cdots, G_{K_j}^j)$ being used on pass j (all probes acting on the same $M + 2$ wires). The user may specify a maximum K' in which case the GA we construct will ensure that $K_j \leq K'$ for all candidate solutions (this may be done

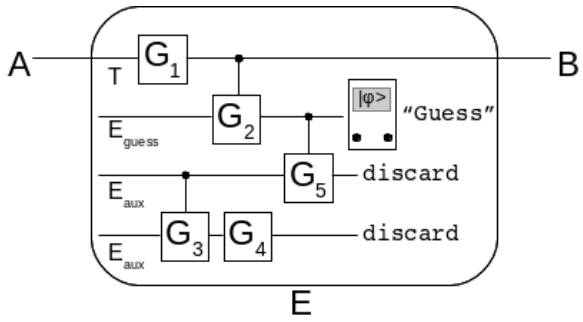

Figure 2: The attack model we consider from [2]. A practical adversary can only use a quantum memory for a very short period of time before a measurement must be made leading E to "guess" at the key-bit being sent. Only shown here is one pass - for multi-pass channels, different circuits attack each pass, however using the same "guess" and auxiliary wires.

Create Gate:	20%
Remove Gate:	30%
Change Wire:	70%
Change Gate Type:	20%
Change Attribute:	80%

Table 1: Showing the probabilities of a particular mutation action being performed. Note that actions are chosen independently of one another and can occur in combinations (thus they need not sum to 100%).

to increase the speed of the algorithm or to represent a resource limited adversary who can only apply a few gates each iteration).

We use a modified form of the crossover and mutation operators introduced in [10]. In particular, cross-over, given two candidate solutions will choose P random cross-over points (where P is the number of passes in the quantum channel). For each of the P lists, the left-most gates from one parent are copied to the child (up to the randomly selected crossover point) while the right-most are copied from the second parent.

To mutate a candidate solution, the system may add a gate (so long as the number of gates is less than the specified maximum K' as discussed); remove a gate; change the wire indices a gate is applied to; change the gate type; mutate an attribute of a gate; or some combination of each. If mutation ever causes an "illegal" operation (e.g., a CNOT gate where $X = Y$), then the gate is treated by the simulator (to be discussed) as the identity gate - i.e., it is ignored. After numerous trials, the probabilities for each of these actions that we chose to use are shown in Table 1; these values produced good results on average in our tests.

Finally, a random solution is produced by choosing random $\{n_j\}_{j=1}^{P}$, where $n_j \leq K'$ specifies the number of gates to add to the attack used on pass j of the quantum channel. Then, for each pass j, n_j random gates (of random types, applied to random wires, and with random attributes) are added to the respective list.

Second Solution Representation - For our second representation type, we evolve arbitrary unitary operators $\{U_i\}_{i=1}^{P}$ (which *could* be constructed out of a suitable collection of primitive gate types if K' is large enough). While this second method has less flexibility (in that users cannot specify the number or types of gates to use), it potentially is able to find attacks breaking the protocol's security at a lower noise level.

These operators act on the transit wire T, Eve's (E's) "guess" wire and an auxiliary system of arbitrary, user specified, dimension n. Thus, each U_i is a $4n \times 4n$ unitary matrix where U_1 is applied on the first pass of the qubit; U_2 on the second, and so on. Following P passes of the qubit, the guess wire is measured in the Z basis and the auxiliary system is discarded; thus U_i must incorporate an appropriate strategy so as to increase E's chance of guessing correctly after this measurement, and after the loss of the auxiliary system (something that was not considered in prior work [6]).

We use the representation method from [5] which discussed the evolution of unitary matrices which cause an input state $|\psi\rangle$ to evolve to a given target state $|\phi\rangle$ (i.e., a unitary U was evolved so that $U|\psi\rangle = |\phi\rangle$). In our case, we wish to evolve unitary operators which, when interacting with a given QKD protocol, will minimize the observed disturbance in the channel, while maximizing E's information gain on the raw key (i.e., maximize the probability she correctly "guesses" at A and B's key-bit). In this representation, a unitary matrix U of dimension $N \times N$ may be decomposed into smaller unitary matrices requiring, in total N^2 real variables. Let $E^{(i,j)}(\phi, \psi, \chi)$ be an $N \times N$ matrix, which has ones along the diagonal, and is zero everywhere else except for:

$$E^{(i,j)}_{i,i} = e^{i\psi}\cos\phi \qquad E^{(i,j)}_{i,j} = e^{i\chi}\sin\phi$$
$$E^{(i,j)}_{j,i} = -e^{-i\chi}\sin\phi \qquad E^{(i,j)}_{j,j} = e^{-i\psi}\cos\phi.$$

with $\phi \in [0, \pi/2]$ and $\psi, \chi \in [0, 2\pi]$. Now, define matrices $E_1, \cdots, E_{N-1}$ as:

$$E_m = \prod_{i=0}^{m-1} E^{m-i,m+1}(\phi_{m-i,m+1}, \psi_{m-i,m+1}, \delta_{i,m-1}\chi_{m+1}).$$

where $\delta_{i,j} = 1$ only if $i = j$ (and 0 otherwise). Finally, $U = e^{i\alpha}E_1E_2\cdots E_{N-1}$ (with $\alpha \in [0, 2\pi]$). A candidate solution, therefore, using this representation is, for each pass of the quantum channel $i = 1, 2, \cdots, P$, a list of four arrays ϕ^i, ψ^i, χ^i, α^i of size $\frac{1}{2}N(N-1)$, $\frac{1}{2}N(N-1)$, $N-1$, and 1 respectively. Crossover will, for each i and for each array individually, choose a random cross over point, placing the left portion of that array from one parent into the child and filling the right portion with the second parent (since α^i is an array of size 1 this means simply choosing randomly to inherit one or the other parent's value for this variable). Mutation will iterate through all variables in all arrays and, with 10% probability, alter that element by a randomly chosen

number in the range $[-\pi, \pi]$ (we do not enforce any boundary conditions on the elements; since the equations are cyclic it would not matter). A random solution is simply a random choice for all elements of the arrays needed to construct $\{U_i\}_{i=1}^{P}$. Finally, for efficiency purposes, our implementation will, after any modification, construct and cache the resulting unitary operators.

3.2 Simulating a Protocol

To simulate the quantum system, we use the quantum simulator developed initially in [6, 7] which was specifically designed to handle the combination of quantum cryptography and evolutionary computation. In particular, the simulator stores density operators as linked-lists of `KetBra` structures, each of which encodes a quantity of the form:

$$\alpha \left| i_1, \cdots, i_n \right\rangle \left\langle j_1, \cdots, j_n \right|,$$

with $\alpha \in \mathbb{C}$ and $i_k, j_k \in \{0, 1\}$. A list of these `KetBra` structures represents a summation of them (and thus an arbitrary density operator). We extended this simulator to support gate based quantum circuits allowing for the application of standard quantum gates in sequence.

For any arbitrary QKD protocol, A and B each have access to certain, private wires. There is at least one "transit" wire T which carries a qubit (if higher-dimensional systems are required, then multiple wires may be used). Any classical public communication can also be simulated using a quantum wire (in practice, such communication would run over a classical communication line; however, it may be modeled as a quantum line also). So long as the qubits are not placed in a superposition, but remain as computational basis states, there is no mathematical difference between this and a true classical line. Finally, E has access to quantum wires private to her.

Abstractly, a QKD protocol consists of two functionalities: first a `computeKeyRate` function and, second, a `computeNoise` function. The first should compute Equation 1; the second should compute the noise induced by the attack under consideration. Both functions must simulate the protocol under investigation resulting in the construction of a density operator description ρ_{ABE} (represented as a list of `KetBra` structures). This density operator is a function of the protocol (e.g., what probabilities A sends qubits and what measurements to make) and also E's attack (which is being evolved). Thus, both functions take as input a candidate solution representing E's attack. To run our algorithm, the user must implement these two functions - essentially, to do so, the user must call certain routines in the simulator to describe the quantum communication stage of the protocol and specify where E has an opportunity to attack (our implementation, with full source code available online, has helper functions allowing a user to call a simple `attack` function at the appropriate time; the software will then build the correct density operator given a candidate solution). Thus users of our system do not require advanced training in quantum mechanics making it a more practical solution than other numerical based methods as mentioned earlier.

3.3 Putting it all Together

To use our system, a user must first describe the protocol, in particular implement the functions `computeKeyRate` and `computeNoise`. This may be done in a straight-forward manner using the quantum simulator. In our current implementation, this entails writing a C++ class inheriting an abstract base class `Protocol`. An instance to a `Protocol` is given to the GA.

The GA will then create an initial population of random solutions. We used a population size of 100 in our evaluations. A new population is created using crossover and mutation as described in the previous section (dependent on the representation type). Selection is simple tournament selection with a tournament size of 3. For each new child solution, it is mutated with probability 80% and added to the next generation. Finally, we use elitism keeping the best solution from the previous generation unaltered.

We wish to minimize the induced noise and the key-rate. Ultimately, the goal is to find an attack against the given protocol that causes the key-rate to be low or negative (which implies A and B need to abort as E has too much information) but with a small disturbance. *If such an attack is found, that means any user of this protocol must abort if an error rate of this level is observed* (since, clearly, an attack exists which gives E too much information while inducing the observed noise). This mechanism can also be used to show the insecurity of a protocol. For instance, if a protocol is given and an attack found against it which induces no noise and has a key-rate of zero, this implies the protocol is completely insecure. Our fitness function is based on the one used in [6], namely: $\texttt{fit} = p \cdot (r - \tau_r)^2 + (1 - p) \cdot (Q - \tau_Q)^2$, where r is the result of the function `computeKeyRate` and Q the result of `computeNoise`. The value of p can be set by the user - we found $p = 0.55$ to produce good results. Finally, τ_r and τ_Q are a "target" rate and noise value which may be set by the user. In our tests, we use $\tau_R = -.02$ and $\tau_Q = 0$ which generally produced good results. In practice, users may alter these values to get a "tighter" estimated noise tolerance as output (our system can only provide upper-bounds on the noise tolerance).

As the primary goal is to output an estimate of the noise tolerance of the given protocol under these practical attacks, at the start of the algorithm, a value $\mathcal{Q}_{out}$ is set to 1. This variable will be the actual estimate of the protocol's noise tolerance. After each new generation is created, the best solution of that population is considered. If the key rate of the protocol is less than some user-specified tolerance T (when faced with this attack), the protocol is considered to be insecure against this attack. In this case, the noise induced by the attack, Q, is considered; if it is lower than $\mathcal{Q}_{out}$, we set $\mathcal{Q}_{out} \leftarrow Q$. That is, we are saving the smallest induced noise, which may be produced by an attack causing the protocol to be insecure (insecurity defined by having a key-rate that is lower than user-specified T). After a user-specified number of generations (1000 in our evaluations below), the value of $\mathcal{Q}_{out}$ is outputted.

Experiment		BB84	six-state BB84	SARG04	B92	SQKD
$\mathcal{G}(1,4)$	Avg	.173	.257	.221	.202	.126
	Min	.154	.211	.205	.174	.103
	σ	.029	.045	.022	.022	.02
	#	17/20	15/20	16/20	16/20	7/10
$\mathcal{G}(3,4)$	Avg	.172	.26	.228	.225	.167
	Min	.154	.211	.206	.194	.167
	σ	.025	.04	.022	.031	10^{-17}
	#	20/20	14/20	13/20	15/20	7/10
$\mathcal{U}(1)$	Avg	.159	.215	.189	.134	.131
	Min	.157	.211	.183	.124	.122
	σ	.002	.006	.004	.006	.006
	#	20/20	20/20	20/20	20/20	10/10
$\mathcal{U}(2)$	Avg	.170	.227	.221	.203	.164
	Min	.161	.215	.208	.169	.142
	σ	.004	.005	.01	.032	.011
	#	20/20	20/20	19/20	20/20	9/10
Tolerance from [2]		.154	.204	.175	n/a	n/a

Table 2: Results of our experiments as discussed in the text. Showing the average $\mathcal{Q}_{out}$ output (Avg), minimum output (Min), and standard deviation (σ) over successful trials (#). If the algorithm did not find an attack causing the key-rate to drop below $T = 0$ during a trial, then it was considered unsuccessful and so no output for that trial was produced. Shown for each experiment is the number (#) of successful trials (over which the average and standard deviation are computed). Also shown are noise tolerances found in [2] if such data is available (the last two protocols were never analyzed in this security model - in fact, it is not clear if SQKD even *could* be analyzed using their method).

4 EVALUATION

We evaluate our algorithm on several QKD protocols, namely BB84, six-state BB84, SARG04, and B92 (descriptions of these protocols may be found in the survey paper [11]). We also evaluate a semi-quantum protocol introduced in [3] (denoted SQKD). The first four protocols are one-pass protocols; the semi-quantum one is a two-pass protocol. The first three were analyzed in [2] giving us a comparison case. The last two protocols were not evaluated before, thus they show that our system can be used to easily analyze new protocols of varying complexity. Indeed, to analyze these systems, we did not need to perform any mathematical analysis - instead we simply inherited a base class, and implemented the two critical functions `computeKeyRate` and `computeNoise`; these were implemented with simple calls to the quantum simulator (full source code available online).

For all evaluations we used a population size of 100. Mutation rates and tournament size are as described in the previous section and remained fixed. The value for τ_Q and τ_R (used in our fitness function) also remained fixed at 0 and -0.02 respectively for all trials.

For each of the five protocols, we ran several independent experiments, each experiment consisting of 20 independent trials (10 for the semi-quantum protocol - due to this protocol's complexity, each trial took significant time to complete; we suspect, however, this can be greatly improved through a more efficient software implementation of our algorithm), each trial consisting of 1000 generations. The output of a single trial was the value $\mathcal{Q}_{out}$ which is the estimated noise tolerance of a protocol after 1000 generations.

To test the gate-based solution representation, the parameters are M, the number of wires allowed by E (the more wires she has, the more temporary quantum processing power she has); and K', the maximum number of gates she can apply in a single pass. We denote this experiment $\mathcal{G}(M, K')$. To test the unitary-based solution representation, the parameters are n, the dimension of E's auxiliary space (the higher this is, the more quantum power she has). We denote this experiment $\mathcal{U}(n)$. Our data is presented in Table 2.

We note that, for BB84, our algorithm finds a noise bound similar to that in [2]. For the other two protocols which were analyzed in that source, our algorithm finds upper-bounds that are close, though higher. Though they are not exact, recall an advantage to our algorithm is more flexibility allowing it to easily analyze other protocols, including multi-pass ones. For B92 and the SQKD protocol, no other data is known so we cannot compare, however the output of our algorithm seems plausible considering the construction of these protocols compared with the other three. An attack against the semi-quantum protocol is shown in Figure 3. The latter is very interesting - for this particular protocol, the key-bit is carried on the first pass connecting A to B. The second pass is only used for error checking. Our algorithm discovered an attack taking advantage of this - namely, E's guess depends only on the forward channel attack; the reverse attack simply "undoes" some of the noise by applying a rotation gate. *We*

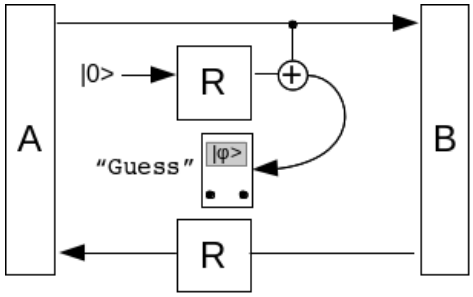

Figure 3: An attack found by our algorithm against the two-way SQKD protocol. This attack induces an error rate of .104 and a key rate of zero. The parameters for the first R gate are $p = .906$, $\theta = 3.789$, and $\psi = 3.804$; for the second R gate (used in the channel connecting B to A) are $p = 1$, $\theta = 2.228$, and $\psi = 3.56$.

did not specifically tell the GA this - we simply inserted the protocol into our simulator leaving it to discover this attack.

Comparing the gate based to the unitary based solution representation we find that the gate based approach is able to find lower noise tolerances in the majority of our test cases (recall, all of these outputs are upper-bounds on the noise tolerances). However, the unitary approach succeeded more often than the gate-based approach where success of a trial is defined to be one where any attack is found causing the key rate to drop below T. Furthermore, the unitary approach had a much lower standard deviation in all cases except one, the $\mathcal{G}(3,4)$ case of SQKD. However, this is due to the fact that the gate based approach in all tests of SQKD in the $(3,4)$ setting found a trivial measure/resend solution. In the $(1,4)$ case, however, the GA found a non-trivial solution in this setting thus causing the noise tolerance bound to be lower. Note, these differences could also be an effect of the fine-tuning of the GA parameters, nonetheless, it can be useful for users of our system to have both gate and unitary methods, and simply take the minimum output of each. Recall that, *any* minimum reported by our system is a valid upper-bound on the noise tolerance, thus one should take the minimum value as the final result of the system.

Mutation probabilities shown in Table 1 were manually fine tuned by running on the BB84 test case since BB84 is a relatively simple protocol and, since we have good bounds on what the value *should* be. Changing an attribute is the easiest thing to modify and we found a high probability of choosing this action was best. Surprisingly we found better results when the mutation removed gates with greater probability than adding gates. Note also that, adding gates, requires additional computation thus slowing the simulation.

5 CLOSING REMARKS

In this paper, we showed how genetic algorithms can be used to study the security of QKD protocols when faced with practical adversaries. We compare two solution representations and evaluated both on five distinct protocols. Our method can be used to easily study new protocols without requiring users to first mathematically convert the protocol into an equivalent entanglement based version (which is not necessarily known how to do for certain multi-pass protocols). It was also able to take advantage of the structure of the protocol to find optimal attacks. Many interesting future problems remain open. Of particular interest would be to adapt this system to allow an adversary to create attacks based on the actual optical instruments used in a protocol. It would also be interesting to consider the effect of noisy operations in E's probe (i.e., if E's gates and measurements were noisy themselves). Ultimately, by having an algorithm to rapidly test security within these various practical security models (which are very difficult to analyze mathematically), this investigation can greatly enhance our understanding of quantum communication and be a useful tool for users of this technology. Our software is available for download at `walterkrawec.org/QKDPracAtk.html`.

REFERENCES

[1] Mustapha Y Abubakar, Low Tang Jung, Nordin Zakaria, Ahmed Younes, and Abdel-Haleem Abdel-Aty. 2017. Reversible circuit synthesis by genetic programming using dynamic gate libraries. *Quantum Information Processing* 16, 6 (2017), 160.

[2] Aurélien Bocquet, Romain Alléaume, and Anthony Leverrier. 2011. Optimal eavesdropping on quantum key distribution without quantum memory. *Journal of Physics A: Mathematical and Theoretical* 45, 2 (2011), 025305.

[3] Michel Boyer, D. Kenigsberg, and T. Mor. 2007. Quantum Key Distribution with Classical Bob. In *Quantum, Nano, and Micro Technologies, 2007. ICQNM '07. First International Conference on*. 10–10. https://doi.org/10.1109/ICQNM.2007.18

[4] Imre Csiszár and Janos Korner. 1978. Broadcast channels with confidential messages. *IEEE transactions on information theory* 24, 3 (1978), 339–348.

[5] S. R. Hutsell and G. W. Greenwood. 2007. Applying Evolutionary Techniques to Quantum Computing Problems. In *IEEE Congress on Evolutionary Computation (CEC 2007)*. 4081–4085.

[6] Walter O Krawec. 2016. A genetic algorithm to analyze the security of quantum cryptographic protocols. In *Evolutionary Computation (CEC), 2016 IEEE Congress on*. IEEE, 2098–2105.

[7] Walter O Krawec, Michael G Nelson, and Eric P Geiss. 2017. Automatic generation of optimal quantum key distribution protocols. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 1153–1160.

[8] M.A. Nielsen and I.L. Chuang. 2000. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, MA.

[9] Stjepan Picek and Marin Golub. 2011. On evolutionary computation methods in cryptography. In *MIPRO, 2011 Proceedings of the 34th International Convention*. IEEE, 1496–1501.

[10] Ben IP Rubinstein. 2001. Evolving quantum circuits using genetic programming. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, Vol. 1. IEEE, 144–151.

[11] Valerio Scarani, Helle Bechmann-Pasquinucci, Nicolas J. Cerf, Miloslav Dušek, Norbert Lütkenhaus, and Momtchil Peev. 2009. The security of practical quantum key distribution. *Rev. Mod. Phys.* 81 (Sep 2009), 1301–1350. Issue 3. https://doi.org/10.1103/RevModPhys.81.1301

[12] L. Spector. 2004. *Automatic Quantum Computer Programming: A Genetic Programming Approach*. Kluwer Academic Publishers, Boston, MA.

[13] Lee Spector and Jon Klein. 2008. Machine invention of quantum computing circuits by means of genetic programming. *AI EDAM* 22, 3 (2008), 275–283.

[14] Xiaoxiao Wang, Licheng Jiao, Yangyang Li, Yutao Qi, and Jianshe Wu. 2015. A Variable-Length Chromosome Evolutionary Algorithm for Reversible Circuit Synthesis. *Journal of Multiple-Valued Logic & Soft Computing* 25, 6 (2015).