Deep reinforcement learning for key distribution based on quantum repeaters

Simon D. Reiß[*] and Peter van Loock[†]
Institute of Physics, Johannes-Gutenberg University of Mainz, Staudingerweg 7, 55128 Mainz, Germany

(Received 20 October 2022; accepted 2 June 2023; published 6 July 2023)

This work examines secret key rates of key distribution based on quantum repeaters in a broad parameter space of the communication distance and coherence time of the quantum memories. As the first step in this task, a Markov decision process modeling the distribution of entangled quantum states via quantum repeaters is developed. Based on this model, a simulation is implemented, which is employed to determine secret key rates under naively controlled, limited memory storage times for a wide range of parameters. The complexity of the quantum state evolution in a multiple-segment quantum repeater chain motivates the use of deep reinforcement learning to search for optimal solutions for the memory storage time limits, typically referred to as memory cutoffs. The main contribution in this work is to explore very general cutoff strategies which dynamically adapt to the state of the quantum repeater. An implementation of this approach is presented, with our focus on four-segment quantum repeaters, achieving a proof of concept of its validity by finding exemplary solutions that outperform the naive strategies.

DOI: 10.1103/PhysRevA.108.012406

I. INTRODUCTION

In the past century, the discovery and progress of quantum physics fundamentally reshaped the understanding of the world. In the 21st century, the scientific community started to utilize quantum mechanics to develop new technologies emerging as an entire field of quantum technology. In this field, quantum cryptography is of high relevance and the first technology to be on its way to broader commercialization. Common classical cryptography relies upon computational hardness assumptions to ensure the security of the transmitted information. This is an inherently vulnerable concept, since advances in the computational power can break the condition on which the security was built. Quantum cryptography, however, offers unconditional security based on fundamental laws of physics [1,2]. In recent years, substantial research has been put into realizing practical quantum cryptography. While fiber-based quantum key distribution (QKD) over 400 km has already been successfully implemented and employed [3,4], achieving high transmission rates over longer distances, up to 1000 km and beyond, remains an ambition for the foreseeable future. Complementary satellite-based quantum communication transmitting quantum information through free space is another promising approach [5,6]. Ultimately, global quantum networks [7–10] are expected to be based upon a combination of fiber and satellite links [11,12]. Another interesting option is the concept of twin-field QKD reaching distances beyond 400 km with standard optical fibers [13] and up to about 800 km by using low-loss fibers (0.1419 dB/km), as was recently demonstrated in Ref. [14].

Fundamentally, the secret key capacity of a lossy bosonic channel is $-\log_2(1-\eta)$ [15,16], and so for long-range point-to-point quantum communication it drops linearly with the transmission parameter η [17] and exponentially with the channel distance. The solution is to extend the range of quantum networks to larger distances by means of a quantum repeater [18,19]. Similar to classical communication, the optical fibers used in quantum communication suffer from channel losses that exponentially grow with distance. Quantum repeaters are designed to overcome these losses and preserve transmission rates at long distances by dissecting the communication channel, distributing entangled quantum states over sufficiently short segments, and eventually connecting the elementary links via quantum teleportation (entanglement swapping) [19]. The ongoing development of quantum repeaters has seen significant progress in recent years in the theoretical concepts [18,20–23], proposals for implementations [24–26], and the engineering of the necessary hardware [27–30].

Besides the experimental implementations, suitable strategies operating the quantum key distribution over quantum repeaters have to be developed. This turns out to be a very challenging task, since the complexity of analyzing multiple-segment quantum repeaters grows quickly with the number of repeater stations and hence the distance [23,31].

Memory-based quantum repeaters store intermediate states in quantum memories and are currently the most experimentally feasible approach that is scalable to large distances by concatenating sufficiently many quantum repeater nodes [27,32]. Hence, one task of the operating protocol is to manage and control the quantum states stored in the quantum memories. Physical implementations of today's quantum memories suffer from degradation of the stored quantum states [27,33,34].

There are at least two established, similar approaches to counteract the memory degradation and improve the fidelity of the distributed states at the cost of lower rates. One approach is based upon a memory cutoff where quantum states are

[*]sreiss@uni-mainz.de
[†]loock@uni-mainz.de

discarded when their storage times exceed a chosen threshold [21,33–40]. An approach to simplify the computation is to use a memory buffer instead (sometimes referred to as memory access time) where the generation of initial entangled states between adjacent repeater stations is restarted at fixed times [41,42]. A memory buffer is distinct from a memory cutoff, since two neighboring segments, even when ready, must wait until a predetermined time resulting in an unnecessary dephasing of the states. In the case of a cutoff, states that have waited for any duration below the cutoff are swapped as soon as possible.

The motivation of this work is to optimize the quantum repeater strategies in quantum key distribution tasks. Exact analytical or even numerical optimization approaches often seem computationally infeasible in the treatment of multiple-segment quantum repeaters [36,38,41–44]. In the following we will use the typical terminology of machine learning where strategies are termed policies.

In Ref. [40] a numerical optimization of the cutoff to maximize the secret key rate was presented. In that work the cutoff was optimized per nesting level. The results presented in Ref. [40] used a fixed (doubling) swapping scheme, which differs from the present work where quantum states are swapped as soon as possible. More importantly, the policies presented in our work are more versatile in their ability to dynamically adapt to the state of the repeater chain and hence are significantly more complex to analyze.

On the way towards optimizing large-scale quantum networks incorporating a vast parameter space, methods able to handle this level of complexity remain of particular interest. Reinforcement learning (RL) is a method capable of finding near-optimal solutions to problems where an analytical treatment is infeasible. Deep reinforcement learning (DRL) extends RL by the use of artificial neural networks in order to handle high-dimensional state spaces. In recent years, DRL has made significant advances in optimizing control tasks for problems which where previously unsolvable [45,46]. Notable examples include training a computer to play Atari games from raw game pixels [47] and performing locomotion tasks [48,49]. More recently, the application of DRL methods to complex video games, which start to capture the complexity and continuous nature of the real world, was successfully demonstrated [50–52]. This motivates the application of these methods to quantum communication networks, offering solutions even for scenarios where the complexity exceeds what is achievable with other numerical optimization approaches.

Most recently, RL was successfully applied to some quantum information tasks. For example, in Refs. [53,54] agents autonomously developed well-known quantum information protocols and completed quantum error correction strategies, respectively. In Ref. [55] RL was used to optimize quantum error correction codes.

The above-mentioned references as well as the present work make use of classical algorithms to solve quantum problems. This should not be confused with quantum machine learning approaches where the optimization itself has quantum aspects to it.

In the present work, DRL will be applied to QKD via quantum repeaters [56]. As the first step, we formulate a memory-based multisegment quantum repeater as a Markov decision process (MDP). This MDP incorporates the full description of the quantum states and incorporates channel loss and Pauli errors as well as the option to discard any intermediate quantum state. Based on this, a simulation is employed, including a simple uniform memory cutoff. We present a broad range of results on the dependence of the secret key rate of the experimental parameters for the memories, the segment lengths, and the uniform cutoff parameter for the special class of four-segment quantum repeaters. The simulation also serves as the necessary groundwork for our DRL approach.

We adapt a public implementation of a proximal policy optimization DRL algorithm to the simulation in order to find sophisticated memory policies optimizing the secret key rate. The major obstacle in this application of DRL is that the optimization merit is nonadditive in terms of the fidelity of the distributed quantum states. We offer an elegant solution in proposing a generalized objective function, expanding common RL algorithms while maintaining the applicability of convergence improving techniques related to value functions. This improves computational feasibility compared to a simple solution via an episodic reward. The search space consists of the full memory control over discarding individual quantum states, based on the entire information available at any moment.

In this dynamic adapting of the policy lies the contribution of our work, extending prior work which considered static fixed cutoffs that were assigned to nesting levels of a doubling scheme or a single point-to-point link. Furthermore, DRL is the enabling method to achieve this versatility. It has proven to excel in optimization tasks whose complexity exceeds the capabilities of other numerical approaches [45,46], thus offering an approach for large-scale networks including numerous interleaving processes and errors. This proves to be an already nontrivial task for the four-segment quantum repeaters as considered here. In principle, the state space of our MDP modeling the quantum repeater is infinite. By setting a maximum accumulated storage time t_m, which will be further defined in Sec. II, one could limit the size of the state space to t_m^9 for a four-segment quantum repeater. Thus, even when assuming t_m as low as $t_m = 10$ the number of states is 10^9. Taking into account that there are at least two possible actions for any relevant state of the MDP and at least two possible transitions for each action, this lower estimation illustrates that analytical approaches are infeasible to solve this optimization problem. The present work can be understood as a proof of concept of a DRL-based optimization method for four-segment repeaters that is equally applicable to larger repeater chains where we expect it to be even more powerful compared with the standard approaches.

Ultimately, we find policies for the quantum memory treatment, which outperform the naive approaches used in the simulations. Therefore, we demonstrate a successful proof-of-concept application of a DRL approach in a first step towards solving complex optimization tasks in quantum networks. The paper is organized as follows. In Sec. II we present our abstract model of a multisegment quantum repeater chain. In Sec. III we describe and discuss the results of simulating four-segment quantum repeaters using this model. In Sec. IV we present our DRL approach and its results, especially in comparison with the simulations without DRL. We

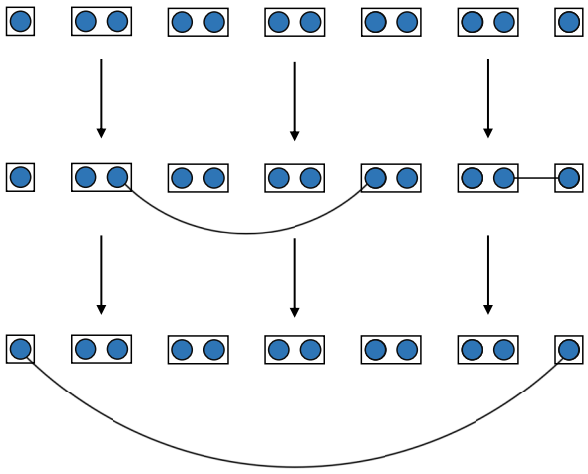

FIG. 1. Multiple-segment quantum repeater chain. Each box depicts a quantum repeater node. Entangled two-qubit states are initially distributed within the segments. As soon as two neighboring segments are ready, i.e., their memory pairs share a successfully distributed state, the entanglement swapping is performed. After subsequent swappings, synchronized via storage at the memory nodes, eventually an entangled state is shared between the two outermost nodes.

summarize in Sec. V. Appendixes A–F provide additional technical details.

II. QUANTUM REPEATER MODEL

This section presents our model of a multisegment quantum repeater including the relevant errors. We will use an MDP to describe the evolution of quantum states in a quantum repeater. We briefly introduce the secret key rate as a figure of merit to evaluate the performance of a quantum repeater. Finally, we will discuss a memory cutoff policy, in which quantum states in the memories of a quantum repeater are discarded in order to improve the fidelity of the states distributed between the communicating parties.

A. Physical model and parameters

Let us now introduce a simplified model of a multisegment quantum repeater. This offers the possibility to obtain fairly general and conceptual results independent of any specific implementation.

A simple, generic multisegment quantum repeater chain is depicted in Fig. 1. Quantum repeaters are used to distribute entanglement between two distant parties by segmenting their connecting channel. Initial entanglement is generated in each segment, for instance, employing a source of entangled photon pairs at a node (or placed in the middle between two memory nodes) and sending one photon to each adjacent node. At each repeater node an entanglement swapping operation, which is essentially a Bell measurement for quantum teleportation, is performed on the memory qubits transferring the entanglement step by step over the entire distance to the communicating parties.

Throughout this work, quantum repeaters with the following properties are considered.

(i) The quantum repeater may in principle consist of an arbitrary number of segments. For simplicity, only one-dimensional concatenated segments are treated here, i.e., quantum repeater chains (an extension to multidimensional repeater networks would also be possible with our methods, but this will not be demonstrated in this work). Eventually, we will focus particularly on four-segment quantum repeaters.

(ii) Each repeater node contains one quantum memory for each adjacent node.

(iii) The repeaters are "clocked repeaters," where the clock times are determined by the classical communication times between nodes.

(iv) All errors of the implementation can be described as Pauli channels,

$$\mathcal{N}(\rho) = \sum_{j=0}^{3} a_j P_j \rho P_j^\dagger, \tag{1}$$

where ρ is the density operator of a quantum state, P_j are the Pauli operators, $\{P_j\}_{j\in\{0,1,2,3\}} = \{\mathbb{1}, X, Z, XZ\}$, and a_j are real-valued non-negative probabilities satisfying the normalization property $\sum_{j=0}^{3} a_j = 1$.

(v) Elements for active entanglement purification [57,58] and more general quantum error correction are not included throughout. The only mechanism to suppress errors on the memories is a finite memory cutoff.

The parameters characterizing the implementation of a quantum repeater in our model are n, the number of segments; L_0, the length of one segment, which is the distance between adjacent repeater nodes; $\nu_i \in [0, 1]$, the probability of any error to occur in a given situation, in particular, the probability of a phase flip to occur on a quantum state stored in a memory; c, the signal speed for classical communication between repeater nodes (typically, the signal speed in an optical fiber, which can be employed for transmitting both quantum and classical signals); and L_{att}, the attenuation length of the optical fibers for realizing the quantum channels between the repeater nodes. The following additional parameters can be obtained from those listed above: $\tau_0 = \frac{L_0}{c}$, one round of quantum and classical communication, which is the time it takes for a node to send qubits and classical information to an adjacent node; τ_c, the memory coherence time for a bipartite quantum state [see Eq. (8)], which corresponds to half of the commonly used parameter of the single-qubit dephasing channel for the quantum memories; $\eta = e^{-L_0/L_{\text{att}}}$, the transmissivity of the optical fibers connecting the repeater nodes, inducing an exponential photon loss with distance; and $p = p_x\eta$, the probability of generating an initial two-qubit entangled state in a segment in one time step. The parameter p_x incorporates the photon creation efficiency of the spin-photon or photon pair source, fiber channel in- and out-coupling, and detector efficiencies, as well as memory write-in efficiencies. The attenuation length L_{att} is a physical parameter of the optical fiber dependent on the employed wavelength of the transmitted photons. The preferred wavelength is that of telecommunication (1.55 μm, potentially requiring wavelength conversions) and a typical value for L_{att} is 22 km. Note that the particular form of the transmissivity η relies on the assumption of constant photon losses, which is the case for optical fibers and this would have

to be changed for different channel implementations (such as fluctuating loss in a free-space channel).

B. Markov decision process

For the purpose of adequately describing a quantum repeater, we designed an MDP to model the propagation and evolution of the quantum states stored in the quantum memories of a multisegment quantum repeater in our physical model. An MDP is defined as a tuple (S, A, P, R), where S is a set of states of the environment, A is a set of actions performed by the agent on the environment, $P : S \times A \times S \to [0, 1]$ is the transition probability in a time step between states dependent on the applied action in this time step, and $R : S \times A \times S \to \mathbb{R}$ is the immediate reward received from a transition.

In practice, besides photon losses, Pauli channels include the most common errors in quantum repeaters and networks [19]. This is fortunate from a mathematical perspective, since Pauli channels commute with the entanglement swapping operations [59,60]. Thus, in the case of memory dephasing, which is a Pauli channel, the number of accumulated error operations is proportional to the accumulated storage time, i.e., additively propagating the storage times of quantum states through the swapping operations gives the correct accumulated dephasing. Therefore, instead of rigorously calculating the density matrix of every intermediate quantum state, it is sufficient to treat the undisturbed quantum states that are only subject to channel loss and count the accumulation of errors separately. Then all these accumulated errors can be applied to the final quantum state distributed between the communicating parties. Hence, the states of the MDP can be encoded in a triangular matrix where each entry corresponds to a pair of repeater nodes storing the accumulated errors of the bipartite quantum state. One time step of the MDP is the time it takes to send quantum and classical information between adjacent repeater nodes (corresponding to τ_0 in our physical model). Note that in our treatment here we only consider the simplest, still to some extent idealized scenario with only two effects determining the final rates: channel loss and memory dephasing. This channel-loss-and-memory-dephasing-only model is similar to that of Ref. [27], however here extended from two to four repeater segments. Additional experimental (error) parameters have been included in the analytical treatment of Ref. [23]. As opposed to the present work, Ref. [23] did not focus on optimizing the memory cutoff (for a discussion on other existing works and approaches for optimizing the memory cutoff, see Sec. II D 1).

The set of possible actions of the agent consists of arbitrary combinations of swapping and discarding operations. The action applied in one time step may perform entanglement swapping on any subset of repeater stations and discard any subset of the stored quantum states. In the event that the two outer nodes of the repeater share an entangled state, this state is discarded and its fidelity is returned as a reward. A detailed description of the MDP can be found in Appendix A.

C. Quantum key distribution: Secret key rate

The secret key rate R is a suitable measure to evaluate the performance of a quantum repeater, as it combines the relevant properties, namely, quantum state fidelity and raw transmission rate, into one convenient figure of merit. Furthermore, long-range QKD is one of the main applications motivating the development of quantum repeaters. The secret key rate is defined as the number of secret bits distributed between two communicating parties, commonly denoted by Alice and Bob, in bits per time and can be written as

$$R = Yr, \tag{2}$$

where Y is the raw rate and r the secret key fraction. The raw rate is the number of raw bits distributed between Alice and Bob per time and the secret key fraction is the potential number of secret key bits that can be extracted per raw bit via post-processing.

Throughout this work the Lo-Chau BB84 [2,61] protocol is used to determine the secret key rates. The asymptotic secret key rate of the Lo-Chau BB84 protocol (see, for example, Ref. [62]) reads

$$R_{\mathrm{BB84}} = Y r_{\mathrm{BB84}}(e_1, e_2), \tag{3}$$

with the secret key fraction

$$r_{\mathrm{BB84}}(e_1, e_2) = 1 - h(e_1) - h(e_2), \tag{4}$$

where h is the binary entropy function defined on the interval $[0, 1]$ as

$$h(p) = -p \log_2 p - (1 - p)\log_2(1 - p). \tag{5}$$

The quantum bit error rates e_1 and e_2 are the probabilities of noncoinciding measurements performed by the two parties on the shared data qubits in the respective coinciding basis. Commonly, the Pauli X and Pauli Z bases are chosen. In a realistic scenario, Alice and Bob have to estimate the bit error rates on a finite set of test data. In the proposal of Ref. [61] it was shown that the number of test bits can be chosen on the order of $\Omega(\log k)$, where k is the length of the final key, while still achieving unconditional security. Thus, the fraction of test data can be chosen asymptotically close to zero. The concrete derivation of the asymptotic secret key rate for our physical model, including only fiber losses and memory dephasing [see later Eq. (8)], can be found in Appendix B and it takes the simple form

$$R_{\mathrm{BB84}} = Y\left(1 - h\left\{\tfrac{1}{2}\left[1 - \mathbb{E}\left(e^{-t/\tau_c}\right)\right]\right\}\right), \tag{6}$$

where τ_c is the coherence time of the quantum memory for a bipartite quantum state and t is the storage time of the quantum state, which is a random variable with an arbitrary but generally unknown probability distribution.

D. Memory dephasing: Cutoff and swapping strategies

1. *Overview of existing approaches*

One of the most significant error sources in practical implementations of memory-based quantum repeaters is the degradation of quantum states in the memories. The distribution of entanglement via a quantum repeater typically is a highly probabilistic process. Thus, the level of degradation caused by memory dephasing can become very large, with some nodes having to wait for others for so long that entanglement can hardly be preserved.

As already mentioned in the Introduction, limiting the time during which the quantum states are stored is a common strategy to improve the fidelity of the distributed quantum states. This concept adapted to quantum repeaters based on imperfect memories was first introduced in Ref. [35]. In terms of resources, this is the least expensive approach to suppress memory errors compared to entanglement purification strategies relying on the distribution of additional entangled-state copies and extra classical communication rounds or, alternatively, strategies based on the implementation of more complicated quantum error correction codes. Again, the processes in a quantum repeater are highly probabilistic and computing the corresponding probability distributions for complex repeater schemes quickly becomes infeasible. In recent years, significant research has been devoted to this and results under various simplifying assumptions have been reported. In order to put our work and results in context, we will give a brief overview of some of the existing literature.

The distributed entangled quantum states in a two-segment repeater including cutoffs were thoroughly analyzed in the presence of various error models in Refs. [27,33,34,59]. Reference [59] also considered more than two segments, however, adapted to specific distribution protocols. In Ref. [37] a closed compact formula was derived to efficiently compute the raw rate of a quantum repeater with an arbitrary number of segments, including a memory cutoff, with the constraint of deterministic entanglement swapping operations and the (practically suboptimal) assumption that all swappings are performed at the end. In Ref. [36] an algorithm based on Markov chains and solving linear-equation systems was presented which exactly computes the average waiting time of quantum repeater chains with an, in principle, arbitrary number of segments including various swapping strategies. For up to four segments, exact rate formulas are given. However, this algorithm is practically limited as its runtime in $O(c^n)$ growth rather quickly with the number of segments n and the cutoff c. In [31] swapping strategies in the presence of nondeterministic swapping operations were optimized for the best possible raw rate, but no cutoff is included.

In a more numerical approach, in Ref. [44] an efficient optimization over entanglement distribution schemes using dynamical programming was proposed under the assumption that there is no time-dependent decoherence in the quantum memories. The algorithm recursively solves larger repeater chains by dissecting them into smaller subchains which are optimized requiring the idealizing simplification that the subprocesses finish at the average time. In Ref. [41] the optimal memory buffer maximizing the rate of distillable entanglement of the average state at all nesting levels in a doubling repeater scheme was computed. In another work, in Ref. [38], algorithms to compute the probability distribution of the fidelity and waiting time for the first distributed entangled state in a quantum repeater and a numerical optimization over a cutoff were presented. Moreover, in Ref. [39] it was shown that the optimal policy maximizing the accumulated fidelity in a multiplexed repeater segment can, in the finite-horizon setting, be computed via a dynamical programming algorithm. In Ref. [42] a heuristic algorithm was developed which optimizes quantum repeater schemes in order to minimize the generation time of an entangled state between communicating parties for a fixed minimum success probability and fidelity. Their schemes also include memory buffers and entanglement distillation. To overcome the computational complexity the schemes are restricted to those that succeed at all levels near deterministically. This is enforced by repeating all probabilistic processes a sufficient number of times to ensure a high probability of at least a single success. In Ref. [63] a genetic algorithm was presented and applied to NetSquid simulations [22], providing insights into the necessary hardware parameters for viable quantum repeaters. Recently, in Ref. [23], an exact rate analysis for quantum repeaters including experimental errors was presented. However, the cutoff was included primarily in a sequential repeater scheme and no optimization of the cutoff was made. Reference [40], as most relevant to the present work, was already discussed in the Introduction.

As was stated in the Introduction, our policies can dynamically adapt to the state of the repeater and decide for any individual quantum state if it should be discarded or not. Thus, we include an extended toolbox which offers the possibility of more sophisticated and, as we will present in Sec. IV E, better policies than previous approaches. In order to handle the complexity introduced by this generalization we use a DRL algorithm. We will distinguish the two policies by referring to the simpler policy as the cutoff policy and to the more sophisticated, better policy as learned policies.

2. *Policies in this work: Cutoff and swapping strategies*

In this paper the cutoff policy is defined such that any quantum state whose accumulated storage time exceeds a chosen cutoff value c will be discarded. Accumulated storage time here refers to the storage time that propagates additively through the swapping operations leading to the final quantum state shared between the most distant stations. If, for example, a state is stored for n_1 time steps and swapped with a state stored for n_2 time steps, the state after the swapping has an accumulated storage time of $n = n_1 + n_2$ time steps. Note that this differs from other common definitions of the cutoff, which often only use the time the state is stored in the current nesting level. However, as was explained in Sec. II B, our accumulated storage time does in fact correctly describe the propagated quantum state. This substantiates the choice of an accumulated cutoff, since it determines the quality of a quantum state more accurately than when only the storage time in one nesting level is taken into account. Also, note that a cutoff per nesting level was employed in Ref. [40], where it was found that for the presented example parameters a nonuniform cutoff, i.e., a different cutoff for each nesting level, does not yield a significant improvement of the secret key rate. Our best guess is therefore that this would also hold for our very similar repeater model. However, a meaningful direct comparison to the numeric results of Ref. [40] is difficult, since their error model assumes depolarizing errors as opposed to our dephasing model. Furthermore, their doubling swapping scheme differs from our swap as soon as possible strategy. Later in this section we argue that our swapping strategy is ideal for our repeater model, giving a further advantage to the secret key rates computed in this work. These arguments justify our uniform cutoff model to be used as a reasonable benchmark.

The cutoff policy as defined in our work, based on an accumulated storage time, actually ignores the fact that the information on which the decision to discard quantum states is based might not be directly available at the node performing the operation at the corresponding time. Our basic assumption that the necessary information is indeed available is not obvious and may even seem unreasonable. Also, an argument that this serves as a bound is invalid, as classical communication is not an obstacle one could possibly circumvent. First, one should note that this does not physically contradict anything about how the quantum repeater is operated and functions. The only but clearly idealizing element of this is that decisions for the controlled part of the process, namely, the discarding of the quantum states, are generally based on the entire state of the MDP. This means decisions of the policy performing an action in one node might be based on information that cannot be possibly accessible in this particular node. In the context of examining fully realistic quantum repeaters, all necessary classical communication must be taken into account. However, in this work, in order to allow for optimal comparability between unlearned and learned policies, we choose these to be all-knowing to give the algorithm at any time complete information to discover new policies. Therefore, imposing the same conditions on the unlearned policies, simulated in the following section, as those imposed on the learned policies, described later, ensures that any advantage of the learned policies must be based on their better strategy and will not be based on any better assumptions.

Next we want to briefly discuss swapping strategies. As a simplification, throughout this work we assume the entanglement swapping to be error-free and deterministic. In the case that memory dephasing is the only error acting on the quantum states, the ideal swapping strategy is to perform the Bell measurement at any repeater node immediately when both adjacent segments have successfully distributed entanglement. This follows from the fact that a swapping operation reduces the number of bipartite quantum states that are simultaneously stored and thus the accumulated dephasing. (For a rigorous and more mathematical treatment of these aspects, together with a systematic formalism to exactly calculate the secret key rate in such protocols that deterministically swap as soon as possible, see Ref. [23]. Among the fastest repeater schemes giving the highest raw rates, i.e., those with parallel entanglement distributions, the analytical proofs of Ref. [23] include and even go beyond a repeater size of four segments as used in our models; when sequential entanglement distributions are also allowed, there is an optimality proof for three identical segments and numerical evidence for up to eight segments.)

All schemes simulated in this work are what we call overlapping schemes. This means that, opposed to other common analysis which often only considers the first distributed state, we consider the generation of many distributed states in an overlapping fashion. This further means that in the lower nesting levels states will be continued to be generated for future processing while other memories are still occupied for the next distributed state. In the four-segment case treated in this work, this might not have a significant effect on the performance of the repeater, as the only case where this is actually beneficial is when an entangled state over three segments is shared. In this case, the segment in the middle of this connection can again start entanglement generation attempts.

III. SIMULATING KEY DISTRIBUTION BASED ON QUANTUM REPEATERS

In this section the cutoff policy is examined via simulations with respect to its effect on the secret key rates. The primary purpose of this is to provide benchmarks of achievable secret key rates to be surpassed, for the policies later considered with the RL algorithm in Sec. IV. Before presenting in detail the results of the simulation, we will briefly describe the specific errors included in the simulation, extending the discussion on our physical model and errors from Sec. II A.

A. Errors and imperfections

A realistic quantum repeater is a complex system with numerous parameters. The more realistic effects are included in a simulation, the more meaningful the results become regarding the assessment of a practical quantum repeater. On the other hand, excluding some sources of errors allows for a more focused view of the most important ones. Another way to look at this is that the results considering fewer errors serve as upper bounds for what is achievable.

The focus of this work is on strategies to counteract the dephasing of quantum memories. For many of the commonly used physical realizations of quantum memories, especially those based on solid-state systems, such as color centers in diamonds, spin dephasing is the dominant error [27,64]. Therefore, the only two imperfections that we choose to include are the dephasing of the quantum memories and the finite transmissivity of the quantum channel (while without the latter quantum repeaters would be pointless).

We model the degradation of the quantum states that are stored in the memories as a dephasing channel

$$\mathcal{N}_Z(\rho) = (1-\nu)\rho + \nu Z\rho Z, \tag{7}$$

which we further specify as exponential decay at time t,

$$\mathcal{N}_Z(\rho, t) = \tfrac{1}{2}\big(1 + e^{-t/2\tau_c}\big)\rho + \tfrac{1}{2}\big(1 - e^{-t/2\tau_c}\big)Z\rho Z, \tag{8}$$

where Z is a Pauli operator and τ_c is the coherence time of the quantum memory [33]. Note that the dephasing channel as defined in Eq. (8) is a single-qubit channel. In a quantum repeater, both qubits of a stored entangled bipartite state are subject to this error channel (individually, i.e., locally and independently) prior to entanglement swapping. This is also the reason for the factor 2 multiplied with the coherence time τ_c, which was defined as the coherence time for a bipartite quantum state. The second imperfection included is the photon loss in the optical fibers, which was already described in Sec. II A.

B. Results

This section presents the results of our numerical simulations. In the simulations, the secret key rates for a BB84 protocol are computed. A discussion of the required formulas can be found in Appendix B. Two-segment quantum repeaters have already been analyzed rather thoroughly [33,34] and, being structurally simple, do not offer extra insights for our

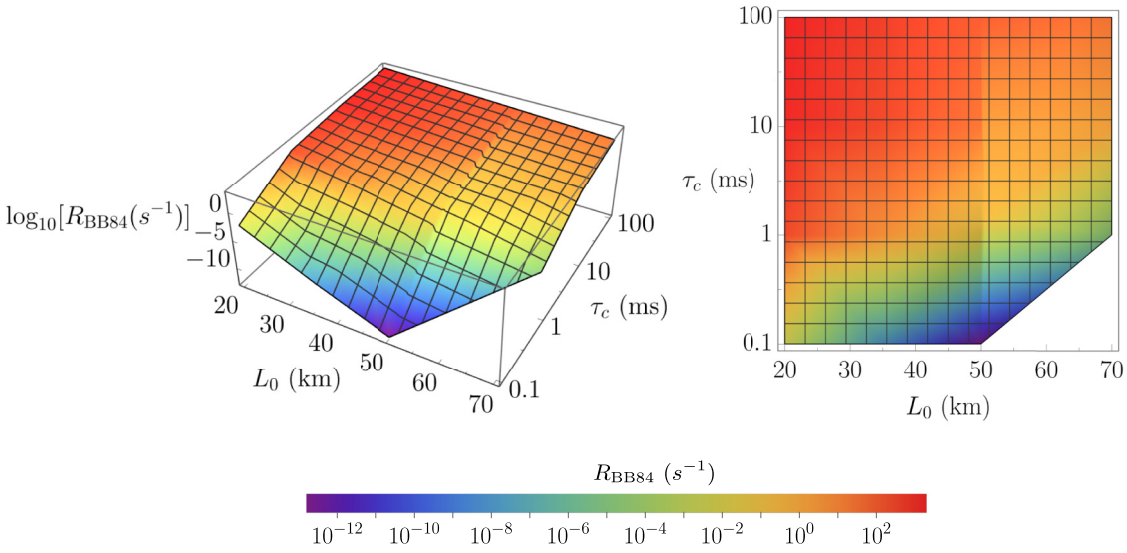

FIG. 2. Secret key rates of four-segment quantum repeaters without memory cutoff. The secret key rate is plotted in secret bits per second dependent on the segment length L_0 and the memory coherence time for a bipartite quantum state τ_c. The secret key rate and the coherence time of the memories are displayed on a logarithmic scale.

treatment. In this work we choose to consider the class of four-segment repeaters. This is partially motivated by the existing literature, which often considers doubling the number of repeater segments, making four segments the logical next step beyond two. Here it is also worth noting that the more segments are included, the less feasible it can become in our model without entanglement purification and error correction that meaningful practical or even nonzero secret key rates can be obtained [23]. For repeaters of a larger scale, such extra tools will have to be included.

For the attenuation length L_{att} we choose 22 km, which is today's experimental standard for the telecom wavelength of 1550 nm [65,66]. The speed of classical communication is assumed to be 2×10^8 m/s, which corresponds to the typical value in optical fibers. The segment lengths are chosen within a commonly considered range of 20–70 km, and the coherence time of the memories is assumed to be within the scope of today's experimental possibilities [27].

1. Uncertainties

In the simulations, since overall minimal uncertainties were achieved, 3-σ confidence intervals (approximately equal to 99.73%) are displayed in all plots with uncertainty bars, instead of the more commonly chosen σ intervals (approximately equal to 68.27%). However, note that the uncertainty bars are still small enough for many data points not to be easily visible, as they can be smaller than the displayed points of the data. One should not be misled by their width in the horizontal direction, since this axis corresponds to a discrete quantity and does not display an uncertainty. The style of displaying the uncertainties was chosen to increase their visibility. Their height corresponds to the 3-σ confidence interval of the secret key rate.

It is also worth noting that the uncertainties get larger for a sparser entanglement distribution in time, thus for longer segment length and smaller cutoff parameters. The reason is that the sparser the received quantum states are, the larger the required sample size is to obtain an equally good estimation of the average fidelity. Hence, the precision of some values with more significant uncertainties is limited by the same computational time needed to simulate larger sample sizes.

2. Quantum repeater without cutoff

In this section the secret key rate of quantum repeaters that are not controlled by a policy including memory cutoff is examined. The results are presented in Fig. 2.

In order to deepen our understanding, we now further elaborate some implications of the parameters. One should note that since the classical communication time $\tau_0 = \frac{L_0}{c}$ scales linearly with the segment length L_0, a larger segment length L_0 not only decreases the transmissivity of the channel, but also decreases the effective coherence time per channel use. Furthermore, an increase in classical communication time results in lower raw rates per second for identical raw rates per channel use, thus reducing the efficiency of a channel use. However, these drawbacks are reasonable taking into account that the overall communication distance increases with the segment length, thus fundamentally lowering the transmission rate.

The results obtained in our simulations are what one would intuitively expect, as the secret key rate increases with smaller repeater segments and better coherence time of the memories. The secret key rate saturates for sufficiently large coherence times, because the fidelity of the distributed quantum states approaches unity, and it drops to zero for short coherence times when the fidelity approaches a value of $\frac{1}{2}$. With longer segment length the maximum secret key rate decreases. The larger the segment length, the higher the requirements on the quantum memories to achieve reasonable secret key rates. The increase of the secret key rates with the quality of the

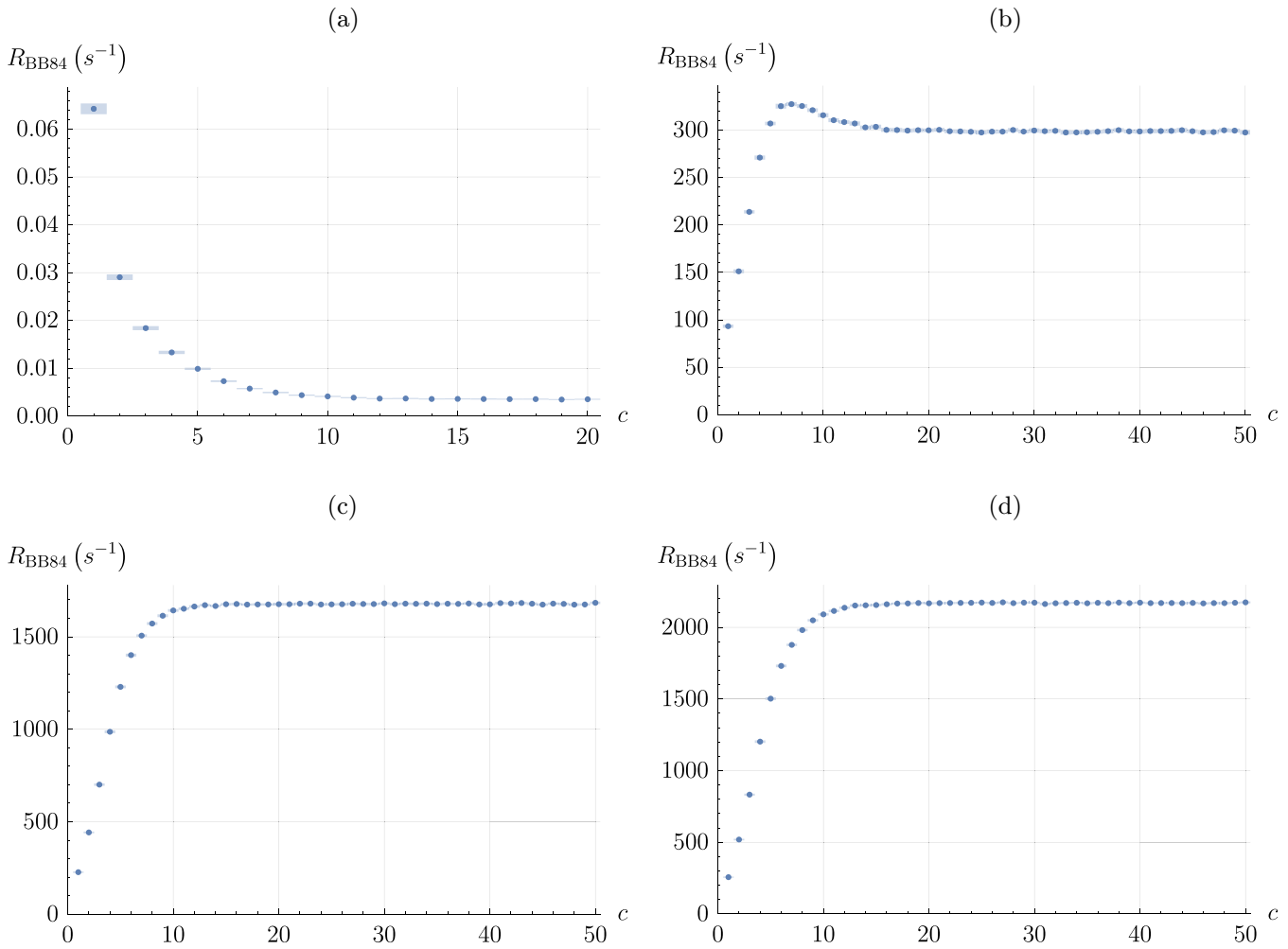

FIG. 3. Secret key rates of four-segment quantum repeaters in secret bits per second dependent on the memory cutoff parameter c for segment length $L_0 = 20$ km and different coherence times of the memories for a bipartite quantum state τ_c, (a) $\tau_c = 0.1$ ms, (b) $\tau_c = 1$ ms, (c) $\tau_c = 10$ ms, and (d) $\tau_c = 100$ ms, plotted with 3-σ confidence intervals.

memories is steeper for worse memories and gets less significant as the secret key rate saturates towards perfect memories. The decrease of the secret key rate with increasing segment distance is significantly steeper for worse memories.

3. Quantum repeater with cutoff

In this section the secret key rates of quantum repeaters that are controlled by the cutoff policy are examined. In Figs. 3–7 the relation between the secret key rate and the cutoff parameter is displayed for various parameter choices of the segment length L_0 and the coherence time of the memories τ_c.

In the limit of large cutoff parameters, the policy is identical to the case without cutoff. This is apparent in all plots, as the secret key rate converges to the no cutoff rates for large cutoff parameters.

For all segment lengths, a similar behavior with respect to the coherence time of the memories is visible. In the regime of low coherence times (i.e., $\tau_c = 0.1$ ms), the secret key rates drop to values near zero, with the ideal cutoff parameter being the lowest possible value, which is one. This observation coincides with the results from Ref. [41]. With increasing coherence time, the optimal cutoff parameter shifts towards larger values, and its peak of the secret key rate for the optimal cutoff decreases relative to the secret key rate of the asymptotically converging quantum repeater without cutoff. This decrease causes the peak to vanish for larger coherence times. At this point, the coherence time reaches a quality value beyond which the cutoff policy does not offer any improvements on the secret key rate.

This behavior moves towards longer coherence times as the segment distance increases. In other words, the shapes of the simulated and plotted secret key rates are qualitatively identical, interpreting the coherence time in relation to the segment length. This indicates that, conceptually, the results when conditioned on the relation between the parameters should be qualitatively applicable to any parameter regime.

In Fig. 8 the ratio between the secret key rate of the best cutoff policy and that without cutoff is shown. This further illustrates the above observations. The advantage of the cutoff vanishes for better quantum memories and shorter segment lengths and it increases for worse

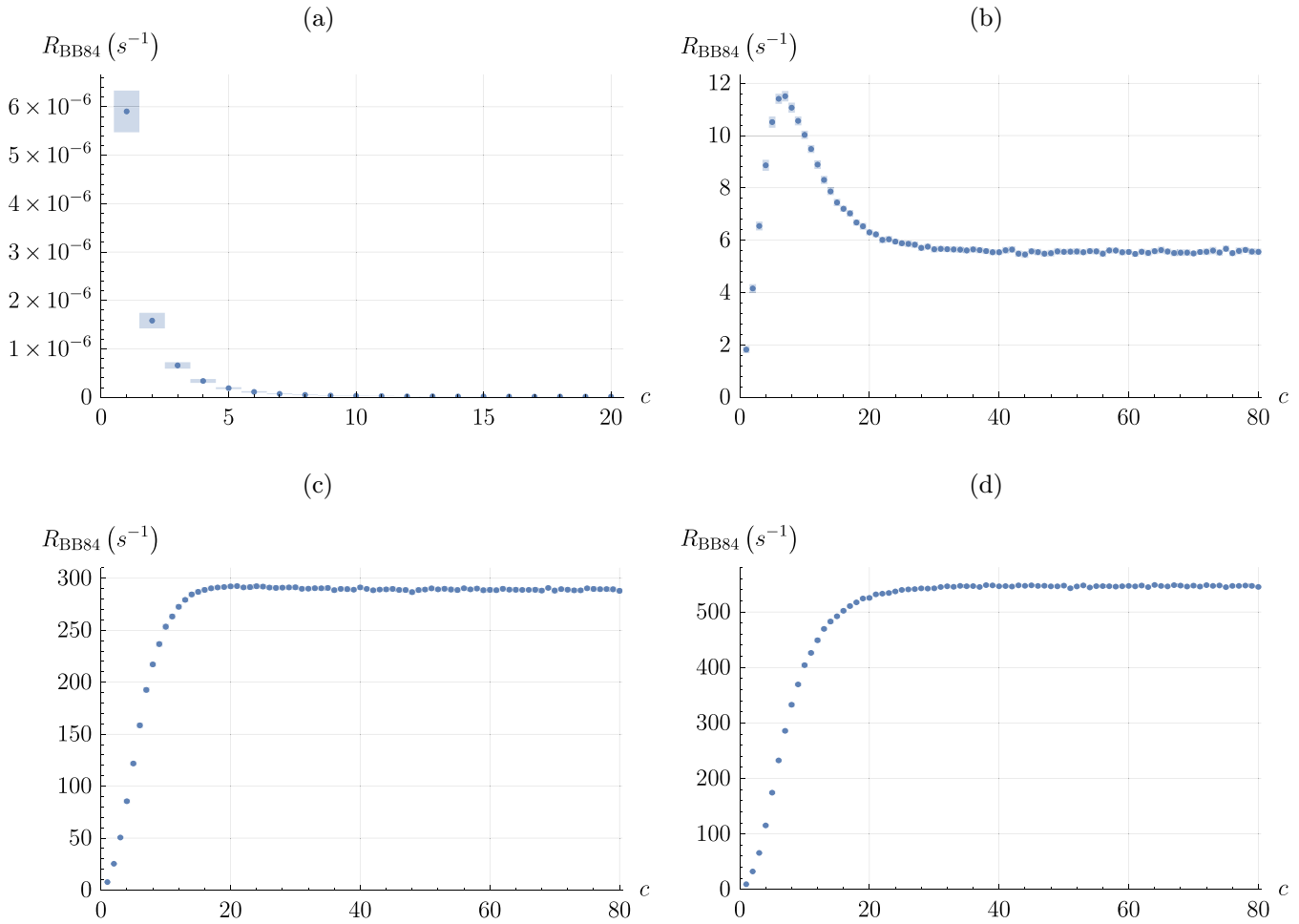

FIG. 4. Secret key rates of four-segment quantum repeaters in secret bits per second dependent on the memory cutoff parameter c for segment length $L_0 = 35$ km and different coherence times of the memories for a bipartite quantum state τ_c, (a) $\tau_c = 0.1$ ms, (b) $\tau_c = 1$ ms, (c) $\tau_c = 10$ ms, and (d) $\tau_c = 100$ ms, plotted with 3-σ confidence intervals.

quantum memories and larger segment lengths. These observations coincide with the results from Ref. [40].

C. Simulation: Conclusion

In this section the secret key rate of the BB84 protocol has been examined via numerical simulations for various parameters of a four-segment repeater setup. The two included imperfections were the dephasing of the quantum memories and the signal attenuation in the optical fibers.

The main observation that can be made with our results is that the worse the coherence time of the quantum memories is in relation to the segment length, the more significant the advantage of the cutoff is. This relation is steeper in the regime of bad memories in relation to the segment length and flattens for better memories until it converges to the point where the cutoff offers no improvement.

IV. DEEP REINFORCEMENT LEARNING APPLIED TO QUANTUM REPEATERS

In this section we present our DRL approach to optimize secret key distribution employing quantum repeaters. The aim is a proof of concept, showing that DRL offers the possibility to provide sophisticated policies for memory treatment which outperform the more naive approaches.

First we will introduce our algorithm and its implementation. For the nonexpert reader this includes a concise self-contained introduction to some DRL theory. The expert reader may skip Sec. IV A and immediately proceed to Sec. IV B. After a brief discussion about the experimentation process, the successful learning runs are presented. For conclusion, some observations about the learned policies are discussed.

A. DRL algorithm

In this section we discuss the algorithm that we have employed. Reinforcement learning is a class of machine learning algorithms which train an agent to optimize its behavior in an environment. This concept is illustrated in Fig. 9. Deep learning uses artificial neural networks to learn hierarchical representations in order to solve classification problems. In DRL these methods are combined by using an artificial neural network to model the agent and approximate value functions [67]. For our DRL application the MDP described in Sec. II B serves as the model of the environment.

There is a vast range of RL algorithms which could be potentially used for our optimizations. It is not possible, even

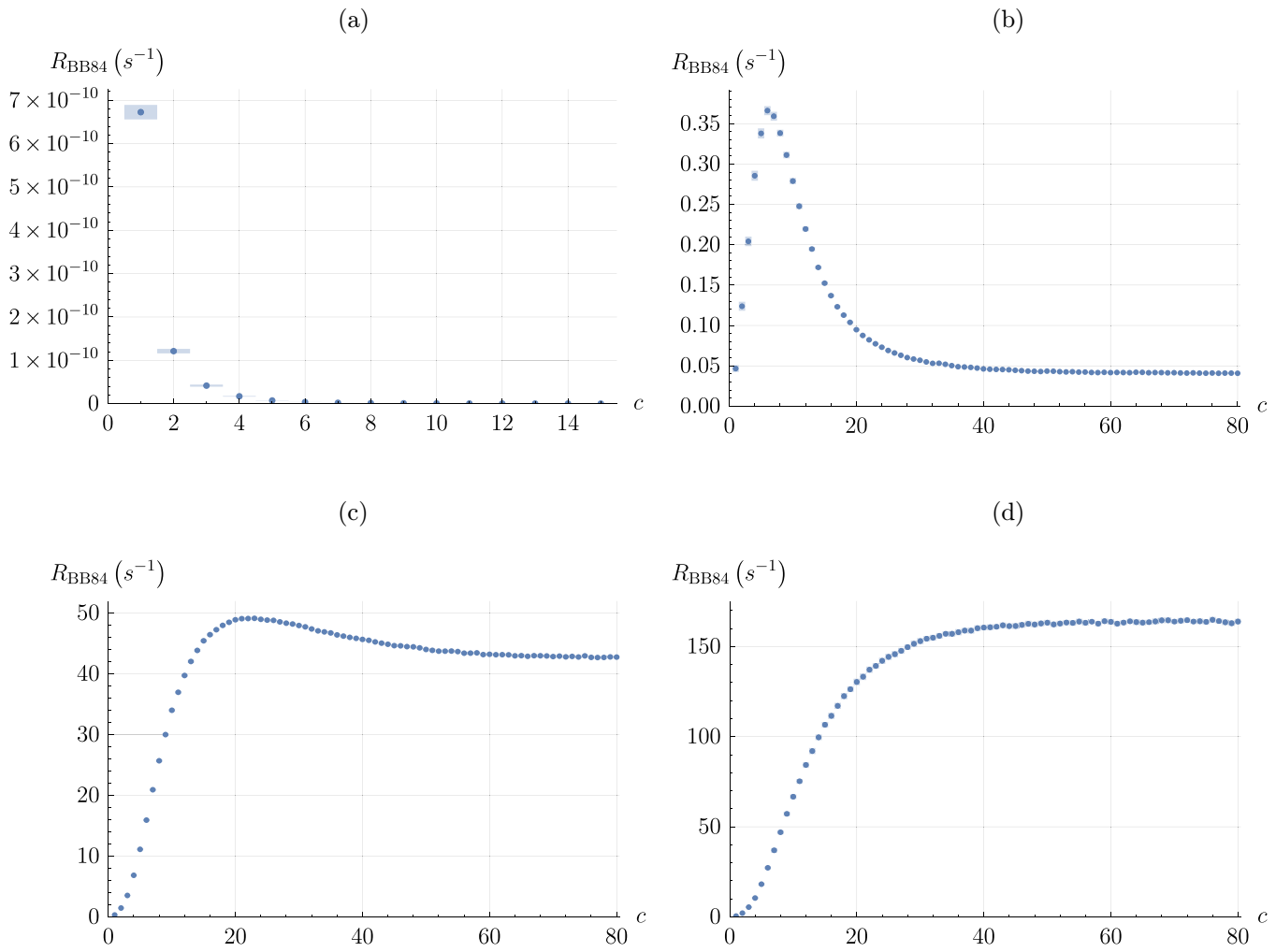

FIG. 5. Secret key rates of four-segment quantum repeaters in secret bits per second dependent on the memory cutoff parameter c for segment length $L_0 = 50$ km and different coherence times of the memories for a bipartite quantum state τ_c, (a) $\tau_c = 0.1$ ms, (b) $\tau_c = 1$ ms, (c) $\tau_c = 10$ ms, and (d) $\tau_c = 100$ ms, plotted with 3-σ confidence intervals.

in principle, to definitely know beforehand what the optimal algorithm for a given optimization problem is. However, we will give some heuristic arguments to motivate our approach. Our choice of a model-free approach over a model-based one is made mainly because of the tendency of the model-free approach to be easier to tune and since model learning is fundamentally hard in complex environments. Model-based approaches are also prone to training an agent performing well in the learned model but badly in the actual environment for the case when the learned model is biased. As a further choice, we perform an on-policy optimization, since this type of optimization tends to be more stable compared with off-policy methods. This, however, is at the cost of sample efficiency [68]. In fact, the chosen approach is well adapted to our application where achieving stable learning is significantly more challenging than efficiently creating large amounts of sample data.

The rest of this section is an introduction to some basic RL concepts, model-free on-policy optimization methods, and the specific algorithm used in this work. For the interested reader Sutton and Barto's book [69] can serve as a comprehensive introduction to RL. A shorter, possibly more practical introduction can be found on the website in [68]. The introduction presented in the following has also taken inspiration from Ref. [68].

We denote the policy of the agent as the probabilistic policy parametrized by a vector θ,

$$\pi_\theta(s, a) : S \times A \to [0, 1]. \tag{9}$$

The policy gives the probability of the agent taking the action $a \in A$ given the environment is in the state $s \in S$. In DRL the agent (i.e., the policy) is encoded in a neural network. In this case, the vector θ consists of the parameters (commonly called weights) of the neural network.

Furthermore, we introduce the common definition of a trajectory as a sequence of states and actions up to a time T called the (time) horizon,

$$\tau = (s_0, a_0, s_1, a_1, \ldots, s_T). \tag{10}$$

The central merit of the optimization is the accumulated discounted reward of a trajectory τ up to a time T,

$$R_0 = \sum_{t=0}^{T} \gamma^t r_t, \tag{11}$$

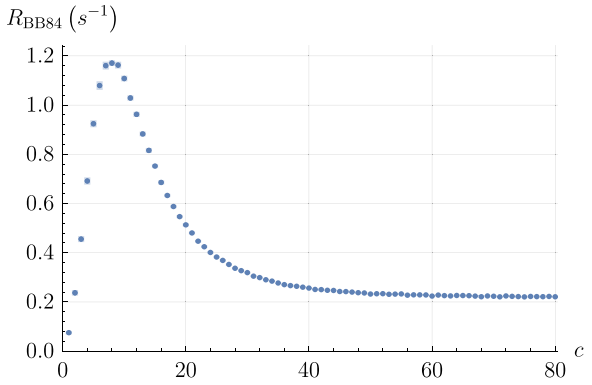

FIG. 6. Secret key rates of four-segment quantum repeaters in secret bits per second dependent on the memory cutoff parameter c for segment length $L_0 = 50.6569$ km and coherence times for a bipartite quantum state $\tau_c = 1.452\,71$ ms, plotted with 3-σ confidence intervals. (These values correspond to the probability to generate initial entanglement in one segment $p = 0.1$ and the dephasing probability in one time step $\nu = 0.08$.)

where γ is called the discount parameter and r_t the immediate reward at time t.

The usual objective of the agent is to optimize the policy π_θ to maximize the expected accumulated discounted reward over trajectories τ,

$$\underset{\theta}{\text{maximize}} \underset{\tau\sim\pi_\theta}{\mathbb{E}}(R_0), \tag{12}$$

where $\mathbb{E}_{\tau\sim\pi_\theta}$ expresses that the expectation value is taken over all trajectories τ following the policy π_θ. Note that the trajectory implies all returned immediate rewards obtained in the trajectory. Further, note that the dependence on γ and T is left out of the expectation value, because it will be fixed in the optimization. These parameters which are fixed in the runtime of the algorithm are termed hyperparameters in order to distinguish them from those parameters that the algorithm optimizes.

Policy gradient algorithms optimize the policy in terms of the objective via a gradient ascent,

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta \underset{\tau\sim\pi_\theta}{\mathbb{E}}(R_0)|_{\theta=\theta_k}, \tag{13}$$

where α is a hyperparameter called the learning rate. This already summarizes the concept of policy gradient optimization. The policy gradient $\nabla_\theta J(\pi_\theta)|_{\theta=\theta_k}$ is estimated on some training data, usually obtained from a simulation, and applied in an iterative fashion. One iteration is typically referred to as an epoch. As this gradient usually cannot be computed exactly and needs to be estimated, this introduces a finite variance into the stochastic gradient ascent. The fact that this gradient can be estimated is not obvious. The proof that this is in principle possible and the computations to do so are beyond what we want to cover in this section and so we refer the interested reader to the aforementioned references [68,69].

Unfortunately, simple DRL algorithms often suffer from slow and unstable convergence properties, i.e., they tend to converge very slowly towards a local optimum and to overshoot in parameter updates, causing them to often drastically lose progress in the learning process. In recent years novel, improved DRL algorithms have been proposed to counteract these issues [45,48,70]. Our algorithm of choice is a proximal policy optimization (PPO) [70], being arguably the most advanced model-free, on-policy DRL algorithm. It has been reported to achieve good results efficiently and in a stable manner [71]. The PPO clip is one of the proposed variants of Ref. [70] and is the one we use in this work. We will now explain the concept behind some of the improvements that were made in these algorithms.

A common approach is to generalize the objective of Eq. (12),

$$\underset{\theta}{\text{maximize}} \underset{\tau\sim\pi_\theta}{\mathbb{E}}(\Phi_t), \tag{14}$$

where we call Φ_t the (generalized) objective function. In general, any objective function Φ_t satisfying

$$\nabla_\theta \underset{\tau\sim\pi_\theta}{\mathbb{E}}(R_0)|_{\theta=\theta_k} = \nabla_\theta \underset{\tau\sim\pi_\theta}{\mathbb{E}}(\Phi_t)|_{\theta=\theta_k} \tag{15}$$

is a valid objective function, since it leaves the gradient in Eq. (12) invariant. Hence, one is free to choose this substitution while still solving the same optimization problem.

In order to discuss how Φ_t will be chosen, we state the standard definitions of the on-policy value function $V^\pi(s_t)$, the on-policy action-value function $Q^\pi(s_t, a_t)$, and the advantage function $A^\pi(s_t, a_t)$.

We define the accumulated reward after time t as

$$R_t = \sum_{l=0}^{T-t} \gamma^l r_{t+l}. \tag{16}$$

Note that this is a generalization of Eq. (11), containing R_0 as a specific case. This is already an improvement, since due to the truncation of the reward in Eq. (16), the evaluation of an action by the advantage function is independent of the trajectory prior to the action. This makes intuitively sense, because the evaluation of an action should only follow from what happened in consequence of that action.

The on-policy value function $V^\pi(s_t)$ is the expected reward received after time t given that the environment at time t is in the state s_t,

$$V^\pi(s_t) = \underset{a_t, s_{t+1}, \ldots, \sim\pi_\theta}{\mathbb{E}}(R_t). \tag{17}$$

The on-policy action-value function $Q^\pi(s, a)$ is the expected reward received after time t given the environment at time t is in the state s_t and the action a_t is taken,

$$Q^\pi(s_t, a_t) = \underset{s_{t+1}, a_{t+1}, \ldots, \sim\pi_\theta}{\mathbb{E}}(R_t). \tag{18}$$

The difference between these functions defines the advantage function $A^\pi(s, a)$, which is a crucial merit in DRL,

$$A^\pi(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(s_t). \tag{19}$$

Therefore, the advantage function evaluates how an action performs relative to the current policy. It can be shown that replacing the discounted reward R_0 in the objective in Eq. (12) by the advantage function leaves the gradient in Eq. (13) invariant [72], as demanded in Eq. (15), i.e.,

$$\nabla_\theta \underset{\tau\sim\pi_\theta}{\mathbb{E}}(R_0)|_{\theta=\theta_k} = \nabla_\theta \underset{\tau\sim\pi_\theta}{\mathbb{E}}[A^\pi(s_t, a_t)]|_{\theta=\theta_k}. \tag{20}$$

To explain the appeal of this substitution, we first note again that in most relevant applications the gradient in Eq. (13) cannot be determined exactly. The solution is to perform

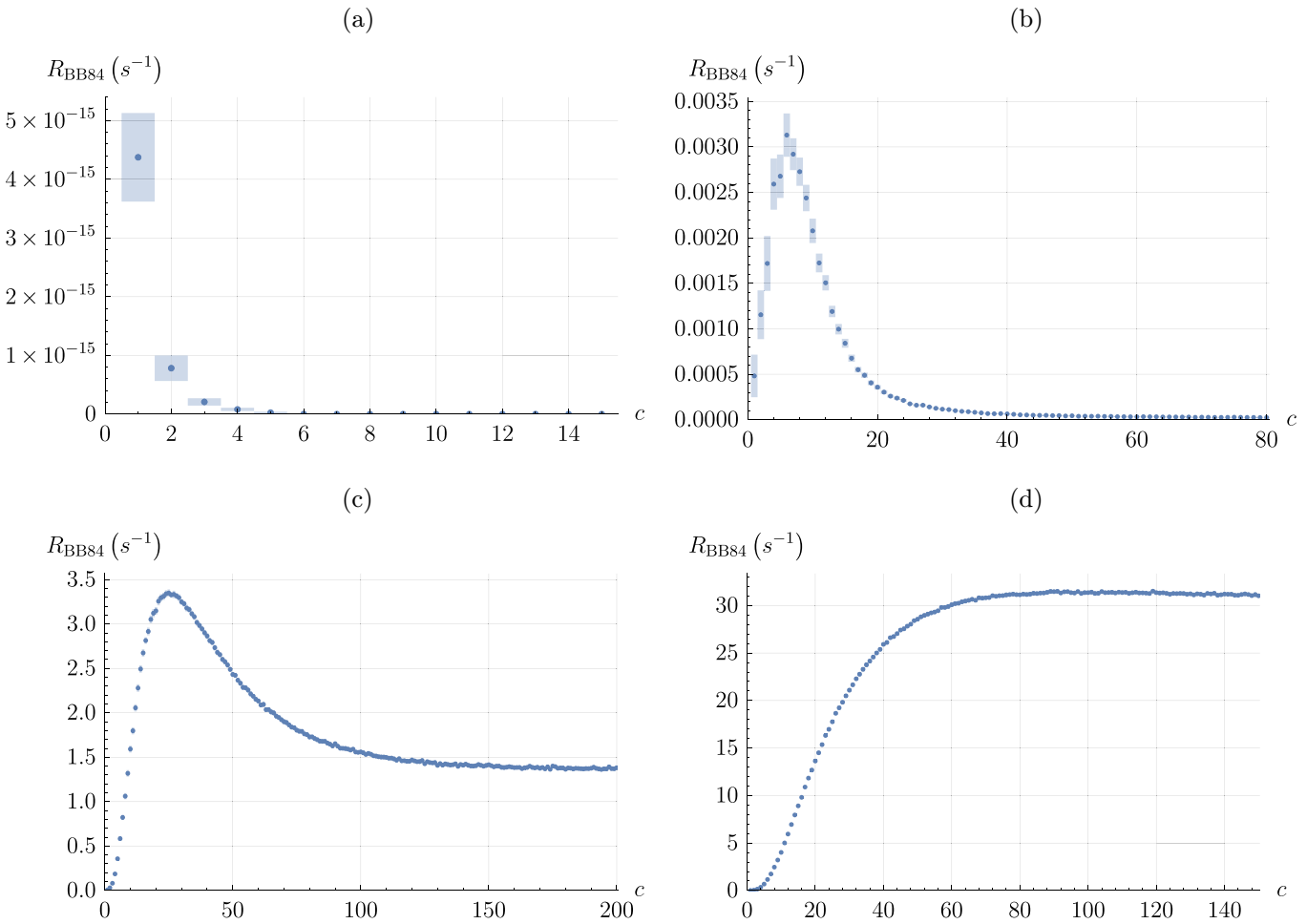

FIG. 7. Secret key rates of four-segment quantum repeaters in secret bits per second dependent on the memory cutoff parameter c for segment length $L_0 = 70$ km and different coherence times of the memories for a bipartite quantum state τ_c, (a) $\tau_c = 0.1$ ms, (b) $\tau_c = 1$ ms, (c) $\tau_c = 10$ ms, and (d) $\tau_c = 100$ ms, plotted with 3-σ confidence intervals.

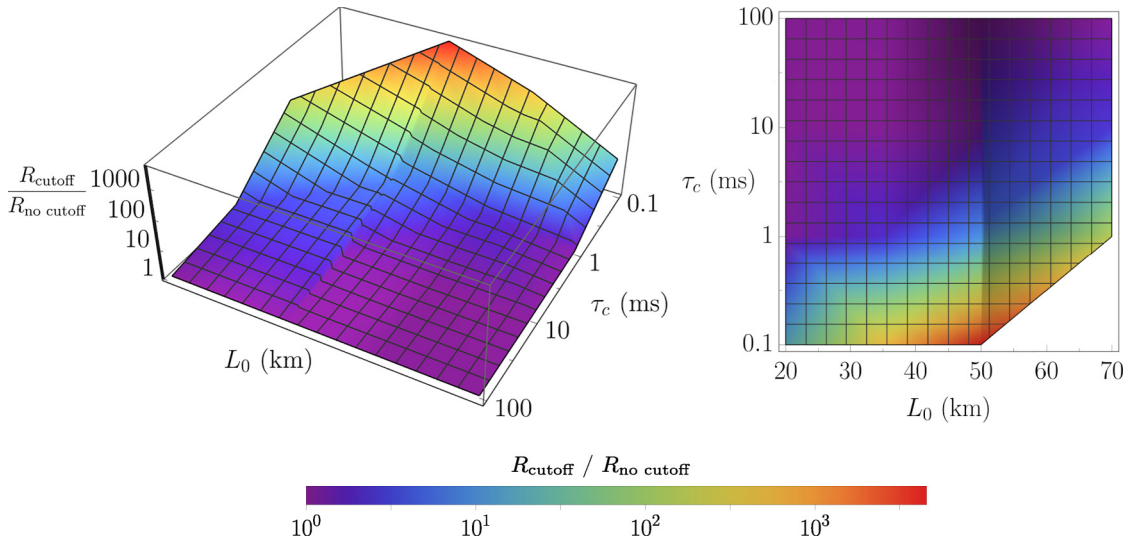

FIG. 8. Simulated ratio between the secret key rates of the scheme with the best memory cutoff policy and schemes entirely without memory cutoff. This ratio and the coherence time of the memories for a bipartite quantum state τ_c are plotted logarithmically.

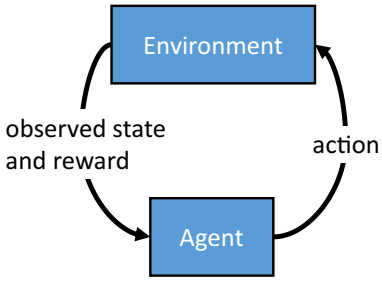

FIG. 9. Conceptual framework of an RL algorithm.

a stochastic gradient ascent, i.e., for each parameter update $\theta_k \to \theta_{k+1}$ the gradient is estimated on a finite sample size. The benefit of using the advantage function in the objective is then to decrease the variance of these estimations, which results in better convergence properties.

This can be intuitively understood by interpreting the advantage function as a rescaling of the reward relative to the expected performance of the current policy. In the case where $Q^\pi(s_t, a_t)$, i.e., the performance merit conditioned on the action a_t, is better than the expectation value over all possible actions $V^\pi(s_t)$, the advantage function is positive; otherwise it is negative. It seems reasonable that we can think of the advantage function as a rescaled reward that becomes positive for any better-than-expected actions and negative for any worse-than-expected actions.

This is just one way in which DRL algorithms such as PPO can attempt to improve their convergence properties. More details can be found in the original PPO proposal [70]. We should also note that the advantage function is just one choice for the objective, as any function satisfying Eq. (15) would be a valid candidate. A detailed discussion on objective functions can be found in Ref. [72].

B. Adaption of the DRL algorithm to nonadditive rewards

We will now discuss how our model of QKD via quantum repeaters can be applied to the DRL approach presented in the preceding section.

A first approach could be to assign the fidelity of an entangled quantum state distributed between the communicating parties to the immediate reward in a time step and return zero reward if no entangled state was distributed in this time step. This approach was, for example, chosen in Ref. [39] to optimize the distributed entanglement in a multiplexed point-to-point quantum communication channel.

In our case, we aim to optimize the secret key rate between the communicating parties. The distributed secret key, however, is not equivalent to the sum of the fidelities of the distributed entangled states. In fact, it is impossible to find an immediate reward function whose sum over time steps evaluates to the secret key rate of an arbitrary trajectory. We prove this in Appendix D. A simple solution consistent with the presented theory of DRL would be to consider an episodic process where the reward of a trajectory is assigned to the last time step. Thus, one could assign the reward as

$$r_t = \begin{cases} 0, & t < T \\ R_{\mathrm{BB84}}(\tau), & t = T, \end{cases} \tag{21}$$

where $R_{\mathrm{BB84}}(\tau)$ is the secret key rate of the distributed quantum states in the trajectory τ. In Eq. (6) it was shown how this can be calculated using the (accumulated) storage time of the distributed entangled quantum states in the trajectory τ. A more detailed explanation of how this is computed using our MDP (described in detail in Appendix A) can be found in Appendix B 2.

The essence of the problem with this approach is often referred to as the credit assignment problem [69,73]. Intuitively, this form of reward includes little information about the causal connection between a specific action and its influence on the reward. Mathematically, this results in an increasing variance of the estimation of the gradient with longer time horizons. Note that the key feature of the advantage function introduced in the preceding section is that it uses a time-grained evaluation of the process. Hence, an episodic reward severely hinders the means with which the convergence of the algorithm is improved. We were not able to achieve good convergence with this approach.

Therefore, in this work we propose an approach where we incorporate a nonadditive reward into the standard mathematical framework of RL. We redefine the optimization objective as

$$\underset{\theta}{\text{maximize}} \underset{\tau\sim\pi_\theta}{\mathbb{E}}[R_{\mathrm{BB84}}(\tau)]. \tag{22}$$

Therefore, the value functions now read

$$V^\pi(s_t) = \underset{a_t, s_{t+1}, \ldots \sim \pi_\theta}{\mathbb{E}}[R_{\mathrm{BB84}}(\tau_t)] \tag{23}$$

and

$$Q^\pi(s_t, a_t) = \underset{s_{t+1}, a_{t+1}, \ldots \sim \pi_\theta}{\mathbb{E}}[R_{\mathrm{BB84}}(\tau_t)], \tag{24}$$

where $\tau_t = (s_{t+1}, a_{t+1}, \ldots, s_T)$ is the trajectory after time step t.

This generalization replaces the accumulated reward after a time step by the secret key rate after that time step. Even though within the scope of this work we did not find a rigorous proof that this fulfills the condition of Eq. (15) and thus is an equivalent optimization, it did deliver well converging results empirically. Based on this approach, we achieved reasonable convergence with good results, which we will present in Sec. IV E. It is important to note that the lack of a rigorous proof for the optimization does not weaken the numerical results obtained, since the performance of a policy is evaluated independent of how the policy is found.

In our implementation, as is common practice, we used an actor-critic function A_t to estimate the advantage function,

$$A_t = R_{\mathrm{BB84}}(\tau_t) - V^\pi(s_t), \tag{25}$$

where an artificial neural network is employed to approximate $V^\pi(s_t)$, which is trained on the same training data the agent is trained on. More details on the implications and properties of our advantage function and on the topic of discounting rewards can be found in Appendix C. Algorithm 1 shows the pseudocode of our implementation. Finally, we would like to stress that generally this adaption could be applied to other optimization problems with nonadditive rewards.

Algorithm 1. PPO algorithm adapted from Ref. [68].

while *not converged* **do**

1. Collect set of trajectories $D_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.
2. Compute reward functions $R_t = R_{\mathrm{BB84}}(\tau|_{t'\geqslant t})$ for every trajectory $\tau \in D_k$ and time step $t \in [0, T]$.
3. Compute $A_t = R_t - V_{\phi_k}(s_t)$.
4. Update the policy by maximizing the RL objective

$$\theta_{k+1} = \arg\max_{\theta} \frac{1}{|D_k|T} \sum_{\tau\in D_k} \sum_{t=0}^{T} \min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A_t, \mathrm{clip}\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}, 1-\epsilon, 1+\epsilon \right) A_t \right) \tag{26}$$

   via Adam stochastic gradient ascent [74].
5. Fit value function by regression on mean-square error,

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{|D_k|T} \sum_{\tau\in D_k} \sum_{t=0}^{T} [V_\phi(s_t) - \Phi_t]^2 \tag{27}$$

   via Adam stochastic gradient descent [74].

C. Application of the algorithm to our physical model

In every time step the agent is given the decision to discard any of the bipartite quantum states stored in the quantum repeater based on the observation of the current state of the environment. It should be pointed out that this allows the agent to develop a vast range of complex policies. The agent is able to make decisions about each quantum state based on an arbitrary selection of the available information.[1] The decision about when to perform a swapping operation is not under the control of the agent. As discussed in Sec. II D, the chosen swapping strategy is fixed to swap as soon as possible.

For the present work, we decided to give the agent at any time the entire information about the current state of the environment. This means that the agent can make decisions on how to operate a quantum memory based on information that physically would not be immediately available at that time for a local operation on the memory. This assumption can be motivated by two primary reasons. The first is that the all-knowing agent can more easily find good solutions. When trying to implement algorithms that are sensitive to their tuned hyperparameters, it is usually a good approach to start with simpler problems and increase the complexity when the first results are obtained. The second reason is that one can possibly learn more about the behavior of the problem within the parameter space when as much information as possible is given into the analysis. This may also be considered as a first step towards methods where an agent is trained to mimic the behavior of the all-knowing agent as well as possible while having only the restricted information of a realistic scenario available. This two-step training of an agent could prove more efficient than trying to learn directly on the realistically restricted information. Finally, it would be highly nontrivial to also include all classical communication in the simulation.

[1]These policies can even emulate other repeater schemes, for example, a sequential repeater scheme like that in Refs. [23,59]. This case is exactly emulated by discarding any quantum states that are generated in segments where the sequential scheme would not attempt any entanglement generation.

The benchmarks for the agents to surpass are the best naive strategies as determined in Sec. III B, which includes schemes both with and without cutoff.

Remark. It is not *a priori* known for which experimental setup the cutoff policy does not provide any benefit. Therefore, there is no reasonable measure for which the no-cutoff approach is as good as the cutoff approach. The cutoff policy can never become worse, as it emulates the no-cutoff approach for large cutoff parameter. Thus, we chose to take the maximum of all data points simulated in Sec. III for each experimental setup as the benchmark. This includes cutoff and no-cutoff strategies. For those cases where the cutoff does not offer an improvement, this means that the approximately optimal cutoff was run multiple times as the secret key rate converged for larger cutoffs, as discussed in Sec. III B. Since from these runs the maximum and not the average was taken, the benchmark is slightly biased towards better secret key rates for these cases. We expect this bias to be negligible for the comparison, since the uncertainties are small enough to prevent significantly overestimated rates.

D. Implementation and experimentation

1. Implementation

The learning algorithm of this work is a PYTHON implementation within the open-source DRL framework OpenAI Spinning Up [68], which uses the Open AI GYM [75]. The Spinning Up framework provides some standard learning algorithms, a logger to store a variety of diagnostics during a learning run, and plotting tools to display them. Our implementation is a modified version of the PPO implementation provided by Spinning Up. Internally, the neural network and the sampling of actions are implemented using the TENSORFLOW [76] library.

The features that had to be modified and added to the implementation of the Spinning Up software are explained in the following.

(a) The calculation of the secret key rate, raw rate, and average fidelity of a trajectory was implemented and included in the main loop of experience collection and storage.

(b) The neural network implemented in the framework did not support multidimensional action spaces. Therefore, the architecture of the neural network and the calculation of the action probabilities π_θ were generalized and adjusted.

(c) The PPO algorithm used a generalized advantage estimation [72], which is incompatible with the nonadditive rewards of QKD. Hence, this was replaced by our generalized actor-critic function of Eq. (25).

The experience collection loop and the Adam stochastic gradient ascent support multiprocess parallelization via Open MPI [77].

The final evaluation of the performance of a learned policy was performed by loading its neural network into the simulation framework of Sec. III. Each agent was evaluated over 100 trajectories, each of length $T = 10^5$. Moreover, a program was implemented to store all the states seen by an agent and the count of actions it took separated for each state. This will be used later to analyze the behavior of the agents.

Remark. The implementation also stores the neural network representing the agent at intermediate epochs. Thus, one could analyze the agent at predetermined points in the learning process.

2. Hyperparameters

The following is a list of the tuned hyperparameters of the implemented algorithm (for details of those parameters which are not discussed in this section, we refer the reader to the original proposal of the algorithm [70] and the documentation of the original implementation [68]): the number of hidden layers, as well as the number of neurons for each hidden layer for the neural networks of the agent and value function estimation (note that the numbers of neurons in the input and output layer are fixed by the state and action space of the environment, respectively); activation functions for each layer of the two neural networks; the horizon of the simulated trajectories; the number of simulated trajectories per epoch; the learning rate for the policy; the learning rate for the value function $V^\pi(s_t)$; ϵ_{clip}, the hyperparameter of the clipped objective of PPO in Eq. (26); the number of gradient steps in the policy update in one epoch; the number of gradient steps in the value function update in one epoch; the maximum Kullback-Leibler divergence which stops the gradient update steps of the policy early if exceeded; and ϵ_{Adam}, a parameter in the Adam optimization to improve numerical stability.

The agent and the value function estimation are represented by a four-layer (output and input layer and two hidden layers) artificial neural network. The hidden layers consist of 32 neurons each. For the activation function of the hidden layers of the agent, the hyperbolic tangent was chosen. The output layer of the agent uses a sigmoid activation in order to interpret the outputs as probabilities. These hyperparameters yielded good convergence properties empirically. The dimension of the input and output layers of the agent corresponds to the dimensions of state and action space of the MDP, respectively. This is 10 and 9 for a four-segment quantum repeater, respectively. The input layer of the value function estimation is also determined by the dimension of the MDP state space and the output dimension is one, thus yielding a scalar value of the value function. The choices for the rest of the hyperparameters are listed in Table IV of Appendix F.

3. Experimentation

The process of the simulated quantum repeaters is highly probabilistic, resulting in high variances for the results of simulated trajectories of short length. This, together with the lack of discounting, leads to slow convergence of the algorithm.

In consequence, hyperparameter tuning turned out to be an extensive task in the setting of this work. The final tuning achieved visible learning progress in the order of an hour, which gives an impression of how time consuming and challenging examining the vast hyperparameter space is where learning progress is significantly less.

With computation time being a major limiting factor and the algorithm being inherently well suited for parallelization, the use of a high-performance computation cluster could very well enable significantly more efficient experimentation.

The TENSORFLOW implementation of the Adam optimizer used was not always numerically stable. That caused the neural network to have "not a number" entries, forcing the learning algorithm to stop. This was counteracted by empirically adjusting the hyperparameter ϵ_{Adam}, which helped but did not solve the problem entirely. As the functionality to resume aborted learning processes was not implemented within the scope of this work, the occurrence of this case was the reason that some of the learning processes presented in this section were not continued, even though they did not converge.

E. Results

The DRL implementation was applied to some of the parameter points of Sec. III B. The neural network of all stored agents of the learning runs as well as the GYM environment simulating the quantum repeaters are publicly available in Ref. [78].

1. Performance and learning progress of the agents

The learned policies can be classified into two categories. In the limiting cases where the naive cutoff offers no improvement on the secret key rate or where the optimal cutoff is one, the learning algorithm approximately reproduced these results. The learning process of these runs is plotted in Fig. 10. In other cases where the naive cutoff offered a significant improvement, the learning algorithm found a policy with even better performance. These cases are plotted in Fig. 11. The list of hyperparameter choices for each run, as well as some discussion on the runtime, can be found in Appendix F.

Tables I and II display the comparison of the learned policies with the benchmark strategies and a no-cutoff strategy.

Table I shows for the first category that the performance of the learned policies is approximately the same as the benchmark. The cases where the benchmarks were matched are exclusively those cases where the parameters correspond to limiting cases identified in Sec. III B, where the optimal strategy is having either no cutoff or the minimal possible cutoff. Therefore, it seems reasonable that in these regimes, there is no more room for improvement for the policies.

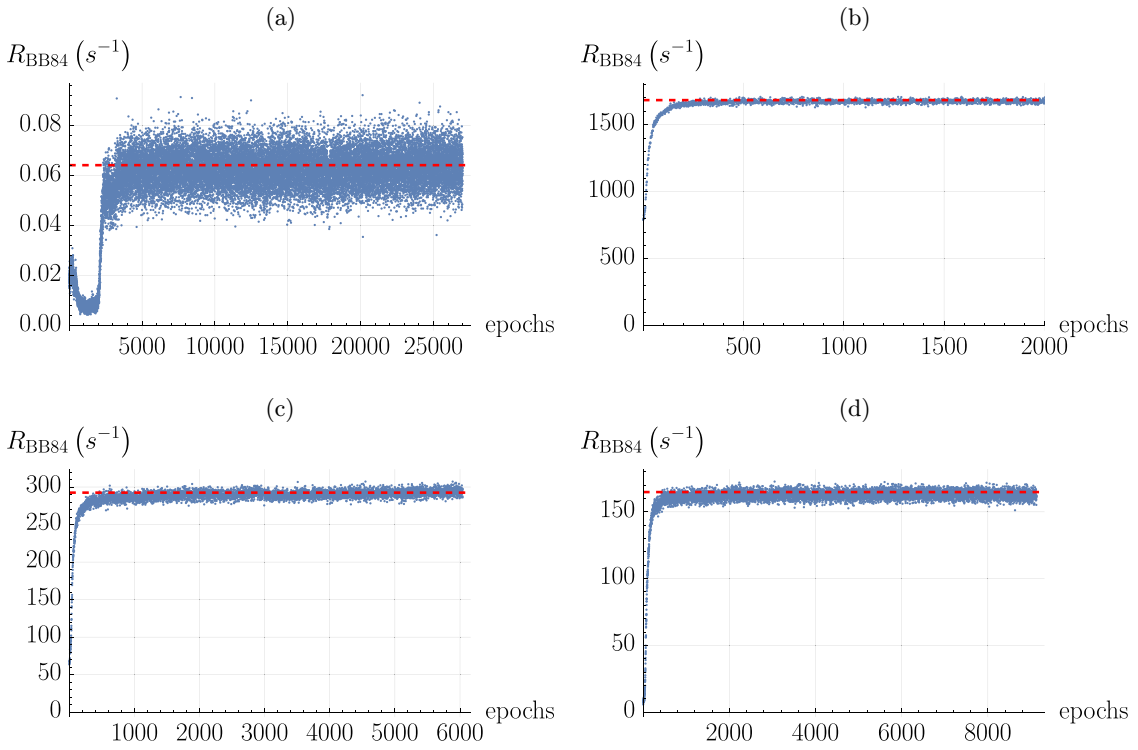

FIG. 10. Learning progress where the performance of the learned policies approximately matched the benchmark marked by the red dashed line. The average over the secret key rate of all trajectories for each epoch is plotted. An epoch is one iteration of the main loop of the algorithm. The runs in (a), (b), and (d) converged without breaking numerically. For these, a long stagnating part after the displayed epochs is not shown. The segment length is (a) and (b) $L_0 = 20$ km, (c) $L_0 = 35$ km, and (d) $L_0 = 50$ km. Here τ_c is the coherence time of the memories for a bipartite quantum state: (a) $\tau_c = 0.1$ ms, (b) and (c) $\tau_c = 10$ ms, and (d) $\tau_c = 100$ ms.

Table II presents the advantage of the learned policies that surpassed the benchmark. The narrow confidence intervals of the ratios indicate that the learned policies outperform the benchmarks. This is further shown via the quantity $\frac{r-1}{\Delta r}$, which is the difference between the measured ratio and the ratio a strategy with no benefit over the benchmark would yield in units of its uncertainty. This therefore prevents the conclusion that the result is a statistical fluctuation. The improvements over the benchmarks range from 2% to 38%. Furthermore, apparently the more improvement the naive cutoff offers over a quantum repeater without cutoff, the more potential lies in the learned policies for this quantum repeater, since this correlation is visible in the achieved and presented examples. It is worth mentioning that this is an analogous result to what was stated in Sec. III C for the naive cutoff policy.

2. *Discussion of the learned policies*

Extracting an in-depth understanding of a policy from the neural network of the agent is a nontrivial task and an open question of current research [79]. Nevertheless, some aspects gained by looking at the policies of the presented results are discussed in this section.

The DRL algorithm optimizes probabilistic policies, where the probability can be interpreted as how certain an agent is about the optimality of an action for a given state. In order to output probabilities, a sigmoid function is applied to the output layer of the neural network representing the policy. The sigmoid function converges asymptotically to one and zero for large absolute output weights. First, this means that in order to get an approximately guaranteed probability, comparably large parameter changes are necessary. Second, a real guaranteed probability can, in theory, never be achieved. This causes even well converged policies to take actions that they evaluate to be nonideal occasionally. This is useful in the learning process, as the agent occasionally reviews these actions, but slightly reduces the final performance of the agent. A suggestion to improve this is to map probabilities above a high threshold to one and analogously for low probabilities to zero in the final policies. Nonetheless, this has not been done in this work.

The actions of those policies that matched the benchmark were confirmed to reproduce their respective benchmark strategy approximately. The agents that surpassed the benchmark are more challenging to comprehend. A complete understanding of the policies could not be achieved in this work. Nevertheless, three observed patterns give insights about how the policies achieved better performance.

(i) The policies keep quantum states with higher storage times shared between nodes that are farther apart compared to quantum states shared between closer nodes, which are discarded much earlier.

(ii) The policy's decision to keep or discard appears to be influenced by the existence and quality of other quantum states in the repeater. If, for example, in the proximity of a quantum state another quantum state exists, the agent is more likely to keep the quantum state.

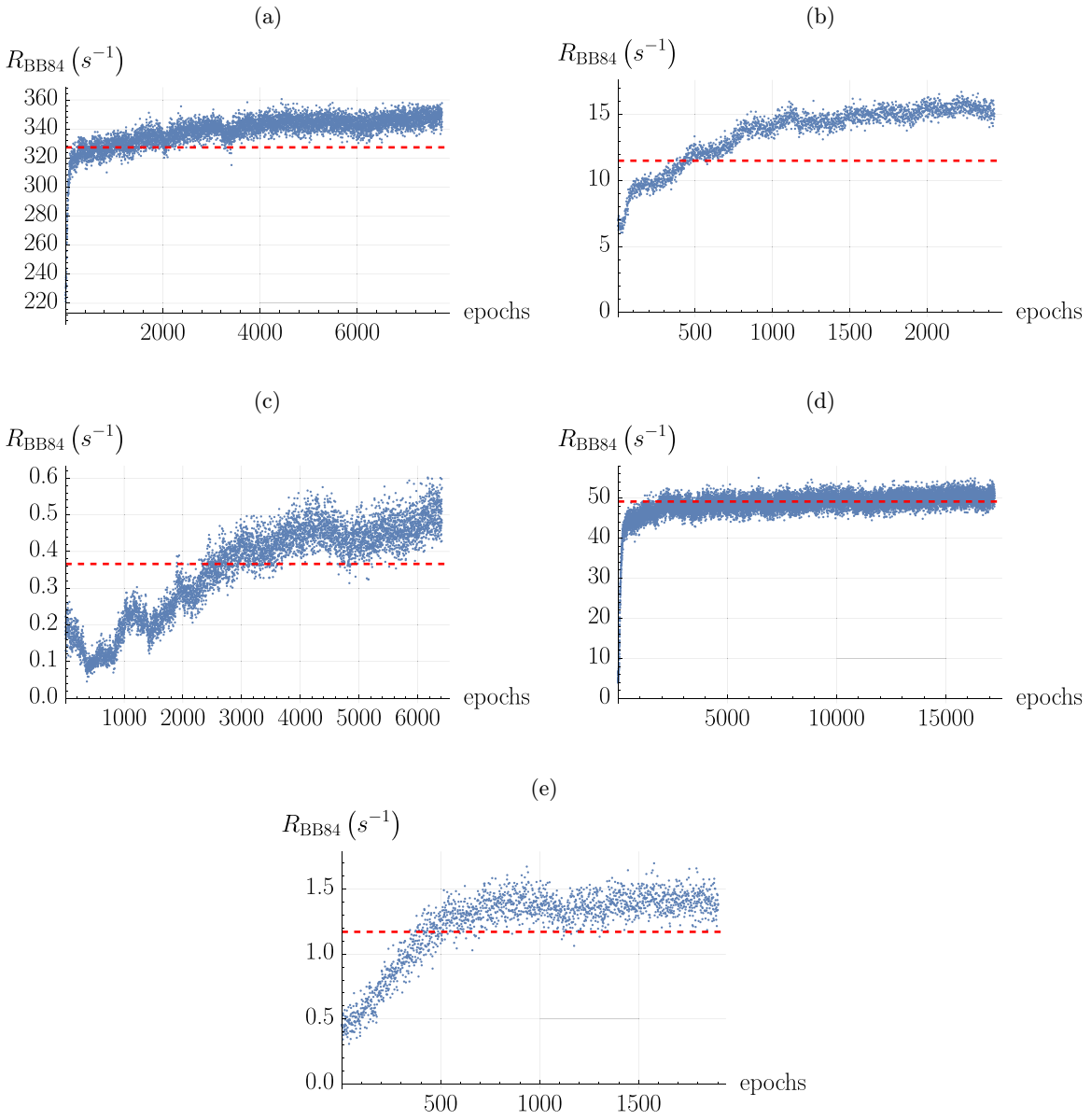

FIG. 11. Learning progress where the performance of the learned policies surpassed the benchmark marked by the red dashed line. The average over the secret key rate of all trajectories for each epoch is plotted. An epoch is one iteration of the main loop of the algorithm. The segment length is (a) $L_0 = 20$ km, (b) $L_0 = 35$ km, (c) and (d) $L_0 = 50$ km, and (e) $L_0 = 50.6569$ km. Here τ_c is the coherence time of the memories for a bipartite quantum state: (a)–(c) $\tau_c = 1$ ms, (d) $\tau_c = 10$ ms, and (e) $\tau_c = 1.45271$ ms.

(iii) The policy discards a quantum state comparably early if a single bipartite quantum state is shared between the two nodes surrounding the first quantum state.

TABLE I. Comparison between the learned policies and the benchmarks for those cases where the benchmark was approximately matched. Here τ_c is the coherence time of the memories for a bipartite quantum state.

Figure	L_0 (km)	τ_c (ms)	$\frac{R_{\mathrm{agent}}}{R_{\mathrm{no\ cutoff}}}$	$\frac{R_{\mathrm{agent}}}{R_{\mathrm{benchmark}}}$
10(a)	20	0.1	18.0 ± 0.2	0.984 ± 0.006
10(b)	20	10	1.001 ± 0.002	0.997 ± 0.002
10(c)	35	10	1.014 ± 0.004	1.001 ± 0.004
10(d)	50	100	0.996 ± 0.003	0.984 ± 0.003

These strategies might seem intuitive or even obvious and could probably be thought of without a machine learning approach. The advantage of the machine learning approach, however, is the scaling of these strategies. Even though the concept might be clear, finding exact numbers remains a task on its own, for which DRL is used here.

On the other hand, some decisions of the agents were obviously unreasonable. This includes decisions of well-converged probabilities. It shows that the full potential of the DRL approach has not been achieved yet in this work and that there is still room for improvement in the algorithm and policies. The following examples are two decisions that an agent made with high probability that fall into this category.

(a) In the case mentioned above, where a policy would discard a quantum state A if it is the only one left in the

TABLE II. Comparison between the learned policies and the benchmarks in the cases where the benchmark was surpassed.

Figure	L_0 (km)	τ_c (ms)	R_{agent} (s^{-1})	$\frac{R_{\text{agent}}}{R_{\text{no cutoff}}}$	$r = \frac{R_{\text{agent}}}{R_{\text{benchmark}}}$	$\frac{r-1}{\Delta r}$
11(a)	20	1	350 ± 8	1.172 ± 0.004	1.068 ± 0.003	20.4453
11(b)	35	1	15.1 ± 0.7	2.67 ± 0.03	1.31 ± 0.01	32.2178
11(c)	50	1	0.51 ± 0.01	12.3 ± 0.1	1.382 ± 0.009	42.9056
11(d)	50	10	50.3 ± 0.8	1.180 ± 0.003	1.024 ± 0.002	10.4957
11(e)	52.6569	1.45271	1.43 ± 0.04	6.45 ± 0.04	1.218 ± 0.005	40.0445

repeater but then keeps it because of another quantum state B in the proximity, some unreasonable actions occurred. In some cases, it kept the state A but discarded the state B, even though the decision to keep state A was dependent on the existence of state B.

(b) In a scenario where two bipartite states with exactly symmetric positions in the repeater where stored, the agent decided to discard the one with the lower storage time and to keep the one with the higher storage time.

Another observation worth mentioning is that the policies tended to be asymmetric in the sense that two symmetric segments were often treated differently. This might happen because of nonideal convergence, but one should note that nothing contradicts the possibility that an asymmetric policy might be optimal, despite the structure of a four-segment repeater being symmetric.

Furthermore, the policies were less converging in their probabilities than their convergence in performance might suggest. This could be explained by the fact that two different actions do not necessarily cause different secret key rates. Some actions might be equivalent or very close in performance. Thus, the agent will not, or will only very slowly, learn to choose one action over the other.

F. Future work

One suggestion for further improvements is based on the fact that states of the environment have significantly different frequencies in a trajectory. Therefore, the agent has superfluous experience about prevalent states of the MDP while having very sparse information about others. This creates an imbalance and thus sample inefficiency. A suggestion to counteract this is to create an environment that uses a random initial state of the trajectory in ways that flatten the distribution of the states the agent sees.

The next step towards analyzing more realistic repeater chains would be to include classical communication by having multiple cooperating agents where each agent has access only to the information available at one corresponding repeater node.

G. Deep reinforcement learning: Conclusion

A proof of concept that DRL can be exploited for QKD via quantum repeaters to find sophisticated policies that improve the secret key rate over naive approaches has been achieved. The results of the learned policies ranged from a reproduction of performance up to a 38% improvement over the naive simulation approach of Sec. III. Furthermore, we have identified shortcomings of the policies and suggested enhancements to the algorithm to further optimize policies, showing potential room for improvement. Additionally, utilizing a high-performance computation cluster would probably go a long way in enhancing the effectiveness with which solutions can be found. This provides motivation to further pursue DRL algorithms in control problems of multisegment quantum repeaters.

The improvements of the learned policies as presented here might seem small for making a strong impact on long-distance QKD applications. This, however, could be misleading, as the examined repeaters only had four segments. Practical applications most likely will use considerably more segments. This increases the number of possible policies and in particular the asymmetry between the repeater nodes. Therefore, we would intuitively expect the improvements to increase for more segments, as the parameter space grows in complexity. Additionally, the complexity of quantum repeaters generally grows with the number of segments, making other approaches less feasible, while DRL is especially suited for problems that can be simulated but no longer analytically analyzed. This emphasizes the meaning of a proof of concept. It has been conceptually shown that DRL can lead to improvements, while the full potential of the approach remains to be seen.

V. CONCLUSION

In this work a Markov decision process to model the generation of spatially distributed, entangled quantum states via a memory-based quantum repeater was developed. In principle, the model can describe quantum repeaters with arbitrary numbers of segments and include arbitrary qubit Pauli and erasure channels as error sources. Moreover, entanglement swapping and discarding of quantum states are actions available to operate the quantum repeater.

Based on this MDP, a simulation was implemented. Among the above-mentioned general error channels, the particular error sources included in the simulation are the random time-dependent dephasing of quantum states in the memories and distance-dependent photon losses in the optical quantum channels (as well as constant losses or inefficiencies at the repeater stations and interfaces). The entanglement swapping is assumed to be error-free and deterministic, and all schemes in this paper swap as soon as possible (which is the optimal strategy in the presence of memory dephasing). The simulation was used to analyze the secret key rate of the BB84 quantum key distribution protocol via four-segment quantum repeaters in a broad parameter space of the segment length and the quantum memories' coherence time. Moreover, the simulation was used to analyze the behavior of the secret key rate under the variation of a controlled limited storage time of

the quantum states, the memory cutoff, in the same parameter space. It was found that there exists a parameter regime in the limit where the memory coherence times are so large or small in relation to the segment length that the ideal control is to discard no or all intermediate quantum states, respectively.

Furthermore, a deep reinforcement learning algorithm was implemented to examine the possibilities of finding sophisticated solutions for the control of the quantum memories improving the secret key rate of quantum repeaters. The algorithm of choice was a proximal policy optimization [70], which was applied to the aforementioned simulated quantum repeaters. First, the results of the limit parameter regime were successfully reproduced. More specifically, and most importantly, the algorithm found policies outperforming the benchmarks provided by the previously employed naive simulation (i.e., a standard simulation without the help of a learning agent), serving as a proof of concept that DRL can indeed offer a valid approach to optimize the memory storage times. This proof of concept is a first step in laying the groundwork to develop and apply DRL algorithms to realistic and practical long-distance quantum key distribution.

The DRL algorithm introduced and employed here, though adapted to the special problem of computing secret key rates in quantum repeaters subject to memory dephasing, is not yet an optimal algorithm. First, it suffers from rather slow convergence properties due to the sparse entanglement distribution per time step for realistic parameters and the lack of an additive reward model. Hence, a next step would be to increase computational power to improve the efficiency with which policies can be found. Another suggestion is to modify the experience loop of the algorithm in a way that flattens the distribution of environment states the agent interacts with to improve the balance of the gradient steps for the different states.

Examining the policies achieved, it is clear that these are not perfect. This suggests that there still is further potential in the approach, even for the examined repeater constellations. In the future, one could expand the simulation to more realistic scenarios, for example, by including more error sources and classical communication, therefore finding solutions for practical settings. In conclusion, the full potential of optimizing policies for quantum repeaters using DRL remains to be seen.

Long-distance key distribution based on quantum repeaters and DRL are both rapidly moving and developing fields, with their full future impact being unforeseeable. This work contributes insights by combining the two fields and showing viability, as well as problems and limitations of the approach.

ACKNOWLEDGMENTS

We thank Frank Schmidt and Evgeny Shchukin for helpful discussions. We thank the BMBF in Germany for support via PhotonQ, Q.Link.X/QR.X, and the BMBF/EU for support via QuantERA/ShoQC.

APPENDIX A: MARKOV DECISION PROCESS MODELING A QUANTUM REPEATER

Here we will formulate the model by explaining each element of the tuple (S, A, P, R).

1. States S

Any state of the Markov model is fully described by the spatially separated bipartite quantum states stored in the quantum memories. Thus, in a general treatment, the full density matrices of all quantum states would be necessary to determine the state of the MDP. By restricting the errors to be Pauli channels as defined in Eq. (1), it is possible to define the states in a much more convenient form. In this restricted scenario, it follows directly from the fact that Pauli channels and entanglement swapping commute that treating the precise development of all quantum states is equivalent to treating a noiseless quantum repeater and applying any contributing errors to the final quantum state. That means it is sufficient to keep track of any error occurring and propagating the counts additive through the swapping. Therefore, all possible states of the MDP of an n-segment repeater can be encoded in a triangular $(n+1) \times (n+1)$ matrix s, with rows and columns corresponding to the repeater nodes. The entries s_{ij} are either the symbol ξ, indicating that the two repeater nodes i and j do not share an entangled state, or a vector with an integer component for each error, storing its accumulated count on the bipartite quantum state.

Generally, the number of possible states is infinite. In realistic scenarios, the memory strategy restricts the transitions between the states in a way that only a finite number of states remain accessible with nonvanishing probability.

2. Actions A

The set of possible actions A can be separated into two independent sets of actions $A = A_s \bigotimes A_d$.

(i) The first set corresponds to entanglement swapping. It is encoded in a vector A_s with a Boolean component for each node. If a component is true, the entanglement swapping operation is performed at that node. Otherwise, no action will be performed at that node.

(ii) The second set corresponds to the discarding of states. It is encoded in a triangular matrix A_d with a Boolean entry for each pair of two different nodes. If an entry is true, the bipartite quantum state of this pair is discarded. Otherwise, no operation will be performed for this quantum state.

It is important to note that in the treatment of the MDP, these actions are instantaneous. The reason for this is that the MDP only models the development of the quantum states, which is independent of classical communication. That does include entanglement swapping. Even though entanglement swapping requires classical communication, the time when classical communication is performed is irrelevant for the development of the quantum states and can thus be excluded from the treatment.

3. Transitional properties P and R

To find a complete analytic expression for P and R is an infeasible task. Fortunately, this is not necessary to simulate the process. It is sufficient to find mathematical operations realizing the simulated processes on the states. One time step of the process is defined as a round of classical communication τ_0, which is the time it takes for a node to send classical information to an adjacent node. This time step is the time

a quantum state will be stored between its initial distribution and the arrival of the classical signals at the corresponding nodes, indicating success. In the following, it will be described how each process that takes place in one time step is simulated.

(1) Initial entanglement is generated in each segment. Within one time step in each segment, an attempt to distribute initial entanglement is performed. This is simulated via sampling in each segment respective to an entanglement generation rate p.

(2) Quantum states are developed in a natural and uncontrolled way: The count of any error that occurs on stored states is increased by one in the tuples in the state matrix s. The relevant example for this work is the dephasing channel.

(3) The following actions are performed.

(a) Swapping at each station j where $(A_s)_j$ is true is performed. For any tuple $\{ik\}$, where $s_{ij} \neq \xi$ and $s_{jk} \neq \xi$, (i) $s_{ik} \leftarrow s_{ij} + s_{jk}$, where the plus sign is the elementwise addition of the two vectors, which follows from the fact that Pauli channels and entanglement swapping commute with each other; (ii) $s_{ij} \leftarrow \xi$ and $s_{jk} \leftarrow \xi$; and (iii) the count of any error that occurs due to entanglement swapping is increased by one in the vector s_{ik}.

(b) The state is discarded: For each tuple $\{ij\}$ where $(A_d)_{ij}$ is true, set $s_{ij} \leftarrow \xi$.

(4) Return $d(s) = s_{0n}$. As was stated in Sec. IV B an immediate scalar reward is not suited for our optimization. In this more general approach we return the entangled state between the outer repeater nodes encoded in the matrix entry $s_{0,n}$ of the state $s \in S$. Formally, we define

$$d(s) = s_{0n}. \tag{A1}$$

In Appendix B 2 we describe how this is used to compute the secret key rate of a trajectory in the MDP.

(5) Since any entangled state between the two outer nodes was used up in the preceding step we set $s_{0n} \leftarrow \xi$.

4. Example step of the MDP

In Fig. 12 one example step of the MDP is illustrated. In this example, the only error is the dephasing of the quantum memories, which accumulates on a state for every time step it is stored. Therefore, the entries s_{jk} are the cumulated storage time of the quantum state stored in the nodes j and k.

The state of the MDP prior to the step is shown in Fig. 12(a). The state after the uncontrolled development, which consists of generating initial entanglement and accumulating the storage time, is shown in Fig. 12(b). The controlled swapping operation is depicted in Fig. 12(c), with the resulting state in Fig. 12(d). The discarding action is displayed in Fig. 12(e), with the resulting state in Fig. 12(f).

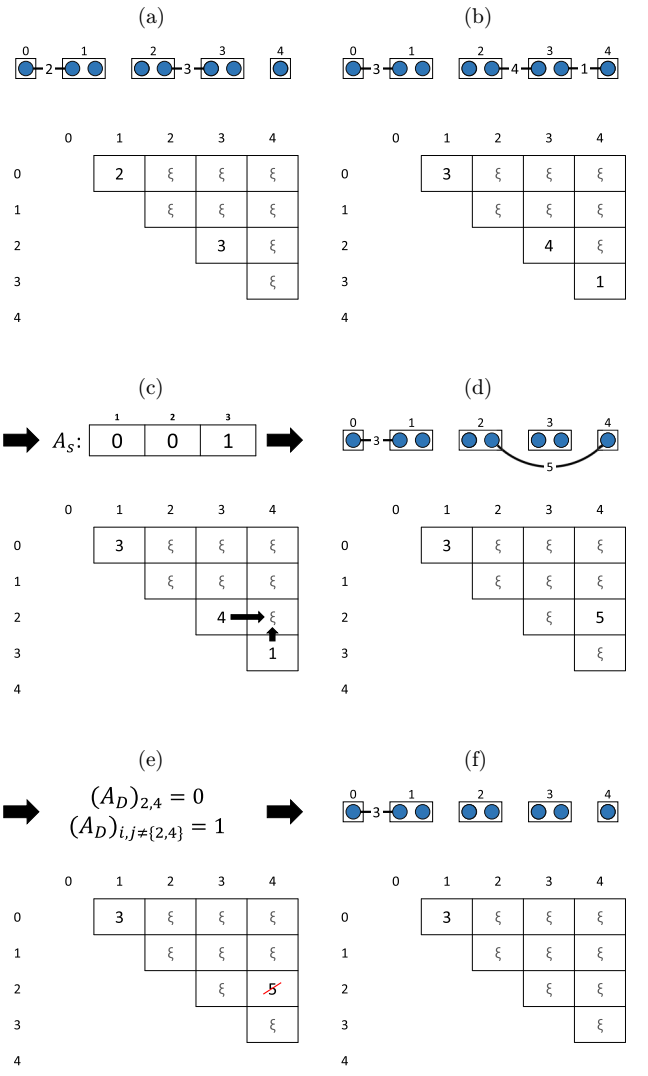

FIG. 12. Illustration of one example step of the Markov decision process.

APPENDIX B: SECRET KEY RATE OF ENTANGLEMENT-BASED BB84 IN THE PRESENCE OF DEPHASING

In the following we will calculate the secret key rate for a BB84 protocol where the quantum states are subject to a dephasing channel. In Appendix B 1 we derive the general expression of Eq. (6). In Appendix B 2 we apply this to a trajectory of the MDP described in Sec. A.

1. General expression

Without loss of generality, we will choose the initially distributed, undisturbed entangled states as $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. The form of a general degraded state follows directly from Eq. (8) and reads

$$\rho = (1-\nu)\,|\Phi^+\rangle\langle\Phi^+| + \nu\,|\Phi^-\rangle\langle\Phi^-|\,, \tag{B1}$$

where

$$0 \leqslant \nu < \tfrac{1}{2} \tag{B2}$$

is the statistical dephasing fraction that depends on a random dephasing time variable with an arbitrary, unknown probability distribution. The bit error rates in the X and Z bases are defined as

$$e_Z = \mathbb{E}(1 - \langle 00|\,\rho\,|00\rangle - \langle 11|\,\rho\,|11\rangle) \tag{B3}$$

and

$$e_X = \mathbb{E}(1 - \langle{+}{+}|\,\rho\,|{+}{+}\rangle - \langle{-}{-}|\,\rho\,|{-}{-}\rangle). \tag{B4}$$

A straightforward calculation yields

$$e_Z = 0 \tag{B5}$$

and

$$e_X = \mathbb{E}(\nu). \tag{B6}$$

Inserting the bit error rates into Eqs. (3) and (4) gives

$$R_{\mathrm{BB84}} = Y\{1 - h[\mathbb{E}(\nu)]\}. \tag{B7}$$

Alternatively, by inserting the exponential dephasing channel defined in Eq. (8), Eq. (B7) takes the form

$$R_{\mathrm{BB84}} = Y\left(1 - h\left\{\tfrac{1}{2}\left[1 - \left(e^{-t/\tau_c}\right)\right]\right\}\right), \tag{B8}$$

where τ_c is the coherence time of the quantum memory for a bipartite quantum state and t the storage time of the quantum state, which is a random variable with an arbitrary unknown probability distribution.[2]

2. Secret key rate of a trajectory

We will now show how the secret key rate $R_{\mathrm{BB84}}(\tau)$ of a trajectory $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T)$ is calculated,

$$R_{\mathrm{BB84}}(\tau) = Y(\tau) r_{\mathrm{BB84}}(\tau). \tag{B9}$$

In the following calculations the function d is defined as in Eq. (A1). This function is used to return the distributed entangled state in this time step. The raw rate $Y(\tau)$ is the number of entangled states distributed between the two outer nodes within the trajectory τ,

$$Y(\tau) = \frac{\sum_{t=0}^{T} \theta(d(s_t))}{T+1}, \tag{B10}$$

where θ,

$$\theta(s) = \begin{cases} 1, & d(s) \neq \xi \\ 0, & d(s) = \xi, \end{cases} \tag{B11}$$

checks if an entangled state state was distributed. Following from Eq. (B8), the secret key fraction reads

$$r_{\mathrm{BB84}}(\tau) = 1 - h\left[\tfrac{1}{2}(1 - \overline{w})\right], \tag{B12}$$

where the expectation value in Eq. (B8) is calculated via the average over the trajectory,

$$\overline{w} = \frac{\sum_{t=0}^{T} \theta(s_t) e^{-d(s_t)\tau_0/\tau_c}}{\sum_{t=0}^{T} \theta(s_t)}. \tag{B13}$$

For the general case with arbitrary Pauli error channels the secret key fraction would read

$$r_{\mathrm{BB84}}(\tau) = 1 - h\left(\frac{\sum_{t=0}^{T} \theta(s_t) e_1\{\rho[d(s_t)]\}}{\sum_{t=0}^{T} \theta(s_t)}\right) - h\left(\frac{\sum_{t=0}^{T} \theta(s_t) e_2\{\rho[d(s_t)]\}}{\sum_{t=0}^{T} \theta(s_t)}\right), \tag{B14}$$

where e_1 and e_2 refer to the two error rates of the bases of the BB84 protocol and

$$\rho(v) = \mathcal{N}_1^{v_1} \cdots \mathcal{N}_D^{v_D}(|\Phi^+\rangle\langle\Phi^+|) \tag{B15}$$

for a vector

$$v = (v_1, \ldots, v_D)^T \in \mathbb{N}^{\mathbb{D}}. \tag{B16}$$

However, this general form of the secret key fraction was not needed for the applications in the present work.

APPENDIX C: DISCOUNTING REWARDS

1. Finite and infinite horizons and the relevance of discounting rewards

The discounted reward function of Eq. (16) introduced in Sec. IV A is a fundamental objective function in RL:

$$R_t = \sum_{l=0}^{T-t} \gamma^l r_{t+l}. \tag{C1}$$

A discounting parameter ν smaller than one introduces the discounting into the accumulated reward. This serves to intuitive purposes. First, it ensures that the objective function remains finite for infinite-horizon trajectories ($T \to \infty$). Second, and maybe most importantly for most practical applications, it makes the agent weight sooner rewards more than rewards farther in the future. This improves the evaluation of an action as it is biased towards its short-term consequences rather than being based on rewards in the distant future, which is increasingly independent from the action at the time t. In this way, discounting reduces noise in the objective function.

As was stated before, the nonadditive rewards in this work are not suited for discounting in this way. Even more detrimental, non-negative rewards can actually decrease the secret key rate of a trajectory. The lack of discounting necessitates the use of a finite-horizon objective function, which makes good gradient estimation difficult. This can be explained by understanding that when the trajectory is too long, the objective function becomes very noisy, since for earlier actions the long trajectory introduces rewards that are mostly independent from early actions. On the other hand, if the trajectory is too short, delayed rewards of an action might not be obtained, causing the trajectory to end, causing a wrong evaluation of the action. In consequence, good convergence is expected to be significantly harder to achieve compared to a problem setting where discounting can be used.

2. Proposal for generalized discounting

In this work an attempt was made to propose a discounted objective function of an arbitrary (not necessarily additive) reward function $\Psi(\tau)$ via

$$\Phi_t = \sum_{l=0}^{T-t} \gamma^l \Psi(\tau_t). \tag{C2}$$

Applied to our optimization, this reads

$$\Phi_t = \sum_{l=0}^{T-t} \gamma^l R_{\mathrm{BB84}}(\tau_t). \tag{C3}$$

[2]In practical finite-size applications, one can use confidence intervals for the estimation of the expectation value to determine an error rate which is larger than the true expectation value with approximately guaranteed probability.

This seemed like a reasonable approach intuitively but yielded no useful results during the minimal testing we did. Therefore, this method was not used further. Even though, unfortunately, this could not be achieved here, with sufficient hyperparameter tuning this method could possibly improve the convergence properties of the optimization significantly, for the reasons discussed above.

APPENDIX D: PROOF SHOWING THE SECRET KEY RATE IS A NONADDITIVE REWARD

In this Appendix we will prove that it is impossible to find an immediate reward function whose sum over time steps evaluates to the secret key rate of an arbitrary trajectory.

To give some intuition, consider the case where many high-fidelity states were distributed between the outer nodes. An additional quantum state of mediocre fidelity might decrease the secret key rate since its increasing of the error rate is more severe than the gain in raw rate. On the other hand, if instead initially very-low-fidelity states are distributed, the same additional mediocre state increases the secret key rate. Therefore, assigning a fixed additive immediate reward independent of the rest of the trajectory is impossible.

Corollary. Assume an MDP (S, A, P, R) as defined in Appendix A. Then there exists no function $R : S \times A \times S \to \mathbb{R}$ such that

$$\sum_{t=0}^{T-1} R(s_t, a_t, s_{t+1}) = R_{\mathrm{BB84}}(\tau)\,\forall\,\tau = (s_0, a_0, s_1, a_1, \ldots, s_T), \tag{D1}$$

where $s_t \in S$, $a_t \in A$, and $R_{\mathrm{BB84}}(\tau)$ is defined in Eq. (B9).

Proof. We will prove this corollary via contradiction. First assume there exists an R which satisfies Eq. (D1). Then, for any trajectory $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T)$,

$$\begin{aligned} R_{\mathrm{BB84}}(\tau) &= \sum_{t=0}^{T-1} R(s_t, a_t, s_{t+1}) \\ &= \sum_{t=0}^{T-2} R(s_t, a_t, s_{t+1}) + \sum_{t=T-1}^{T-1} R(s_t, a_t, s_{t+1}) \\ &= R_{\mathrm{BB84}}(\tau_{\mathrm{trunc}}) + R(s_{T-1}, a_{T-1}, s_T), \end{aligned} \tag{D2}$$

where $\tau_{\mathrm{trunc}} = (s_0, a_0, s_1, a_1, \ldots, s_{T-1})$. We obtain the equation

$$R(s_{T-1}, a_{T-1}, s_T) = R_{\mathrm{BB84}}(\tau) - R_{\mathrm{BB84}}(\tau_{\mathrm{trunc}}). \tag{D3}$$

Now we consider a second trajectory $\tau' = (s'_0, a'_0, s'_1, a'_1, \ldots, s'_{T-1}, a'_{T-1}, s_T)$ which is identical to τ in the last state s_T. Therefore, we again have a relation like

$$R(s_{T-1}, a_{T-1}, s_T) = R_{\mathrm{BB84}}(\tau') - R_{\mathrm{BB84}}(\tau'_{\mathrm{trunc}}). \tag{D4}$$

It is easy to find examples for τ and τ' for which the right-hand side of Eq. (D3) evaluates to a positive value and the right-hand side of Eq. (D4) evaluates to a negative value, leading to the contradiction.

For a simple example, we assume the channel-loss-and-memory-dephasing-only model leading to the distributed quantum state defined by a single variable ν of the form in Eq. (B1). Furthermore, we assume in the trajectory τ, N entangled states were distributed before T, each with dephasing parameter ν, and one entangled state was distributed in the last time step T with dephasing parameter ν_T. Equation (D3) in this example reads

$$\begin{aligned} &R(s_{T-1}, a_{T-1}, s_T) \\ &\quad = \frac{N+1}{T}\left[1 - h\left(\frac{N\nu + \nu_T}{N+1}\right)\right] - \frac{N}{T-1}[1 - h(\nu)] =: R_\nu. \end{aligned} \tag{D5}$$

For the second trajectory τ' we simply assume a different parameter ν' for the entangled states prior to time step T, leading to $R_{\nu'}$ for Eq. (D4). Now we assume $N = 3$, $T = 9$, $\nu_T = \frac{1}{4}$, $\nu = \frac{1}{20}$, and $\nu' = \frac{9}{20}$ and we obtain $R_\nu \approx -0.012$ and $R_{\nu'} \approx 0.014$ and thus $R_\nu \neq R_{\nu'}$. ■

APPENDIX E: TRAJECTORY LENGTHS OF THE SIMULATIONS

In this Appendix the lengths of the trajectories used in the simulations and the final evaluation of the agents are given in Table III.

TABLE III. Length of the trajectories T for different repeater parameters of the simulations with and without cutoff and the final evaluation of the agents.

L_0 (km)	τ_c (ms)	T, no cutoff	T, cutoff	T, agent
20	0.1	10^4	10^4	10^5
20	1	10^4	10^4	10^5
20	10	10^4	10^4	10^5
20	100	10^4	10^4	
35	0.1	10^4	10^4	
35	1	10^4	10^4	10^5
35	10	10^4	10^4	10^5
35	100	10^4	10^4	
50	0.1	10^6	10^6	
50	1	10^5	10^4	10^5
50	10	10^5	10^4	10^5
50	100	10^4	10^4	10^5
50.6569	1.45271	10^5	10^5	10^5
70	0.1	10^5	10^6	
70	1	10^5	10^5	
70	10	10^5	10^5	
70	100	10^5	10^5	

APPENDIX F: HYPERPARAMETERS OF THE DRL RUNS

The DRL program was run with four processes on a desktop computer with a four-core CPU.[3] An epoch with four trajectories, each with 8000 simulated time steps, took about 13 s on average. Therefore, 1000 epochs take around 3.6 h. In the operated hyperparameter space, the computation time was roughly linear with the number of simulated time steps

[3]Intel(R) Core(TM) i7-3770 CPU @ 3.40 GHz.

TABLE IV. Hyperparameters of the learning runs presented in Sec. IV E.

Figure	L_0 (km)	τ_c (ms)	T	N	α_π	α_V	ϵ_{clip}	n_π	n_V	KL	ϵ_{Adam}
10(a)	20	0.1	800	4	1×10^{-4}	1×10^{-3}	0.2	120	80	0.015	0.01
11(a)	20	1	8000	4	4×10^{-4}	1×10^{-3}	0.2	120	80	0.015	0.01
10(b)	20	10	8000	4	1×10^{-3}	1×10^{-3}	0.2	120	80	0.015	0.01
11(b)	35	1	8000	4	1×10^{-3}	1×10^{-3}	0.2	120	80	0.015	0.01
10(c)	35	10	8000	4	4×10^{-4}	1×10^{-3}	0.2	120	80	0.015	0.01
11(c)	50	1	8000	4	8×10^{-4}	1×10^{-3}	0.2	1000	80	0.015	0.1
11(d)	50	10	8000	4	4×10^{-3}	1×10^{-3}	0.2	1000	80	0.015	1
10(d)	50	100	8000	4	4×10^{-3}	1×10^{-3}	0.2	1000	80	0.015	1
11(e)	50.6569	1.45271	8000	4	4×10^{-4}	1×10^{-3}	0.2	120	80	0.015	1×10^{-8}

within the epoch. This serves as an orientation for the required computation time of each run.

The architecture of the neural networks representing the policy and the value function was chosen to be identical for every learning run as it seemed sufficient in complexity and degrees of freedom for the task and smaller architectures did not empirically improve computation time significantly. The architecture consisted of two hidden layers with 32 neurons each. The activation function of the hidden layers was chosen to be the hyperbolic tangent. The neural network representing the policy applies a sigmoid function as the activation on the output layer so that the final output can be interpreted as probabilities.

The following abbreviations are used for the hyperparameters: T, the horizon of the simulated trajectories; N, the number of simulated trajectories per epoch; α_π, the learning rate for the policy; α_V, the learning rate for the value function; ϵ_{clip}, the epsilon of the clipped objective of PPO in Eq. (26); n_π, the number of gradient steps in the policy update in one epoch; n_V, the number of gradient steps in the value function update in one epoch; KL, the maximum Kullback-Leibler divergence which stops the gradient update steps of the policy early if exceeded; and ϵ_{Adam}, the parameter in the Adam optimization to improve numerical stability. The hyperparameters for each learning run presented in Sec. IV E can be found in Table IV.

[1] P. W. Shor and J. Preskill, Phys. Rev. Lett. **85**, 441 (2000).

[2] C. H. Bennett and G. Brassard, Theor. Comput. Sci. **560**, 7 (2014).

[3] H.-L. Yin, T.-Y. Chen, Z.-W. Yu, H. Liu, L.-X. You, Y.-H. Zhou, S.-J. Chen, Y. Mao, M.-Q. Huang, W.-J. Zhang, H. Chen, M. J. Li, D. Nolan, F. Zhou, X. Jiang, Z. Wang, Q. Zhang, X.-B. Wang, and J.-W. Pan, Phys. Rev. Lett. **117**, 190501 (2016).

[4] A. Boaron, G. Boso, D. Rusca, C. Vulliez, C. Autebert, M. Caloz, M. Perrenoud, G. Gras, F. Bussières, M.-J. Li, D. Nolan, A. Martin, and H. Zbinden, Phys. Rev. Lett. **121**, 190502 (2018).

[5] J. Yin, Y. Cao, Y.-H. Li, S.-K. Liao, L. Zhang, J.-G. Ren, W.-Q. Cai, W.-Y. Liu, B. Li, H. Dai, G.-B. Li, Q.-M. Lu, Y.-H. Gong, Y. Xu, S.-L. Li, F.-Z. Li, Y.-Y. Yin, Z.-Q. Jiang, M. Li, J.-J. Jia *et al.*, Science **356**, 1140 (2017).

[6] G. Vallone, D. Bacco, D. Dequal, S. Gaiarin, V. Luceri, G. Bianco, and P. Villoresi, Phys. Rev. Lett. **115**, 040502 (2015).

[7] H. J. Kimble, Nature (London) **453**, 1023 (2008).

[8] S. Wehner, D. Elkouss, and R. Hanson, Science **362**, eaam9288 (2018).

[9] J. Illiano, M. Caleffi, A. Manzalini, and A. S. Cacciapuoti, Comput. Netw. **213**, 109092 (2022).

[10] A. S. Cacciapuoti, J. Illiano, S. Koudia, K. Simonov, and M. Caleffi, IEEE Netw. **36**, 6 (2022).

[11] C. Harney and S. Pirandola, PRX Quantum **3**, 010349 (2022).

[12] C. Harney, A. I. Fletcher, and S. Pirandola, Phys. Rev. Appl. **18**, 014012 (2022).

[13] M. Lucamarini, Z. L. Yuan, J. F. Dynes, and A. J. Shields, Nature (London) **557**, 400 (2018).

[14] S. Wang, Z.-Q. Yin, D.-Y. He, W. Chen, R.-Q. Wang, P. Ye, Y. Zhou, G.-J. Fan-Yuan, F.-X. Wang, Y.-G. Zhu, P. V. Morozov, A. V. Divochiy, Z. Zhou, G.-C. Guo, and Z.-F. Han, Nat. Photon. **16**, 154 (2022).

[15] S. Pirandola, R. García-Patrón, S. L. Braunstein, and S. Lloyd, Phys. Rev. Lett. **102**, 050503 (2009).

[16] S. Pirandola, R. Laurenza, C. Ottaviani, and L. Banchi, Nat. Commun. **8**, 15043 (2017).

[17] M. Takeoka, S. Guha, and M. M. Wilde, Nat. Commun. **5**, 5235 (2014).

[18] S. Muralidharan, L. Li, J. Kim, N. Lütkenhaus, M. D. Lukin, and L. Jiang, Sci. Rep. **6**, 20463 (2016).

[19] H.-J. Briegel, W. Dür, J. I. Cirac, and P. Zoller, Phys. Rev. Lett. **81**, 5932 (1998).

[20] S. Slussarenko and G. J. Pryde, Appl. Phys. Rev. **6**, 041303 (2019).

[21] S. Khatri, C. T. Matyas, A. U. Siddiqui, and J. P. Dowling, Phys. Rev. Res. **1**, 023032 (2019).

[22] T. Coopmans, R. Knegjens, A. Dahlberg, D. Maier, L. Nijsten, J. de Oliveira Filho, M. Papendrecht, J. Rabbie, F. Rozpędek, M. Skrzypczyk, L. Wubben, W. de Jong, D. Podareanu, A. Torres-Knoop, D. Elkouss, and S. Wehner, Commun. Phys. **4**, 164 (2021).

[23] L. Kamin, E. Shchukin, F. Schmidt, and P. van Loock, Phys. Rev. Res. **5**, 023086 (2023).

[24] L.-M. Duan, M. D. Lukin, J. I. Cirac, and P. Zoller, Nature (London) **414**, 413 (2001).

[25] P. van Loock, T. D. Ladd, K. Sanaka, F. Yamaguchi, K. Nemoto, W. J. Munro, and Y. Yamamoto, Phys. Rev. Lett. **96**, 240501 (2006).
[26] L. Childress, J. M. Taylor, A. S. Sørensen, and M. D. Lukin, Phys. Rev. Lett. **96**, 070504 (2006).
[27] P. van Loock, W. Alt, C. Becher, O. Benson, H. Boche, C. Deppe, J. Eschner, S. Höfling, D. Meschede, P. Michler, F. Schmidt, and H. Weinfurter, Adv. Quantum Technol. **3**, 1900141 (2020).
[28] P. C. Humphreys, N. Kalb, J. P. J. Morits, R. N. Schouten, R. F. L. Vermeulen, D. J. Twitchen, M. Markham, and R. Hanson, Nature (London) **558**, 268 (2018).
[29] M. K. Bhaskar, R. Riedinger, B. Machielse, D. S. Levonian, C. T. Nguyen, E. N. Knall, H. Park, D. Englund, M. Lončar, D. D. Sukachev, and M. D. Lukin, Nature (London) **580**, 60 (2020).
[30] S. Langenfeld, P. Thomas, O. Morin, and G. Rempe, Phys. Rev. Lett. **126**, 230506 (2021).
[31] E. Shchukin and P. van Loock, Phys. Rev. Lett. **128**, 150502 (2022).
[32] N. Sangouard, C. Simon, H. de Riedmatten, and N. Gisin, Rev. Mod. Phys. **83**, 33 (2011).
[33] F. Rozpędek, K. Goodenough, J. Ribeiro, N. Kalb, V. C. Vivoli, A. Reiserer, R. Hanson, S. Wehner, and D. Elkouss, Quantum Sci. Technol. **3**, 034002 (2018).
[34] F. Rozpędek, R. Yehia, K. Goodenough, M. Ruf, P. C. Humphreys, R. Hanson, S. Wehner, and D. Elkouss, Phys. Rev. A **99**, 052330 (2019).
[35] O. A. Collins, S. D. Jenkins, A. Kuzmich, and T. A. B. Kennedy, Phys. Rev. Lett. **98**, 060502 (2007).
[36] E. Shchukin, F. Schmidt, and P. van Loock, Phys. Rev. A **100**, 032322 (2019).
[37] L. Praxmeyer, Reposition time in probabilistic imperfect memories, arXiv:1309.3407.
[38] S. Brand, T. Coopmans, and D. Elkouss, IEEE J. Sel. Areas Commun. **38**, 619 (2020).
[39] S. Khatri, Policies for elementary link generation in quantum networks, Quantum **5**, 537 (2021).
[40] B. Li, T. Coopmans, and D. Elkouss, IEEE Trans. Quantum Eng. **2**, 4103015 (2021).
[41] S. Santra, L. Jiang, and V. S. Malinovsky, Quantum Sci. Technol. **4**, 025010 (2019).
[42] K. Goodenough, D. Elkouss, and S. Wehner, Phys. Rev. A **103**, 032610 (2021).
[43] S. E. Vinay and P. Kok, Phys. Rev. A **99**, 042313 (2019).
[44] L. Jiang, J. M. Taylor, N. Khaneja, and M. D. Lukin, Proc. Natl. Acad. Sci. U.S.A. **104**, 17291 (2007).
[45] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, IEEE Signal Process. Mag. **34**, 26 (2017).
[46] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, in *Proceedings of the 33rd International Conference on Machine Learning, New York, 2016*, edited by M. F. Balcan and K. Q. Weinberger (JMLR, Cambridge, 2016), Vol. 48.
[47] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, Nature (London) **518**, 529 (2015).
[48] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, in *Proceedings of the 32nd International Conference on Machine Learning, Lille, 2015*, edited by F. Bach and D. Blei (JMLR, Cambridge, 2015), Vol. 37.
[49] N. Heess, G. Wayne, D. Silver, T. Lillicrap, Y. Tassa, and T. Erez, *Proceedings of the 29th Conference on Neural Information Processing Systems, Montreal, 2015* (Curran, Red Hook, 2015).
[50] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg *et al.*, Nature (London) **575**, 350 (2019).
[51] W. H. Guss, C. Codel, K. Hofmann, B. Houghton, N. Kuno, S. Milani, S. Mohanty, D. P. Liebana, R. Salakhutdinov, N. Topin, M. Veloso, and P. Wang, in *Proceedings of the 33rd Conference on Neural Information Processing Systems, Vancouver, 2019*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran, Red Hook, 2019).
[52] OpenAI C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. d. O. Pinto, J. Raiman, T. Salimans, J. Schlatter *et al.*, Dota 2 with large scale deep reinforcement learning, arXiv:1912.06680.
[53] J. Wallnöfer, A. A. Melnikov, W. Dür, and H. J. Briegel, PRX Quantum **1**, 010301 (2020).
[54] T. Fösel, P. Tighineanu, T. Weiss, and F. Marquardt, Phys. Rev. X **8**, 031084 (2018).
[55] H. P. Nautrup, N. Delfosse, V. Dunjko, H. J. Briegel, and N. Friis, Quantum **3**, 215 (2019).
[56] S. D. Reiß, Applying deep reinforcement learning to key distribution based on quantum repeaters, M.Sc. thesis, Johannes Gutenberg University, 2019.
[57] C. H. Bennett, H. J. Bernstein, S. Popescu, and B. Schumacher, Phys. Rev. A **53**, 2046 (1996).
[58] C. H. Bennett, G. Brassard, S. Popescu, B. Schumacher, J. A. Smolin, and W. K. Wootters, Phys. Rev. Lett. **76**, 722 (1996).
[59] F. Schmidt and P. van Loock, Phys. Rev. A **102**, 042614 (2020).
[60] G. Bowen and S. Bose, Phys. Rev. Lett. **87**, 267901 (2001).
[61] H.-K. Lo, H. F. Chau, and M. Ardehali, J. Cryptol. **18**, 133 (2005).
[62] S. Pirandola, U. L. Andersen, L. Banchi, M. Berta, D. Bunandar, R. Colbeck, D. Englund, T. Gehring, C. Lupo, C. Ottaviani, J. Pereira, M. Razavi, J. S. Shaari, M. Tomamichel, V. C. Usenko, G. Vallone, P. Villoresi, and P. Wallden, Adv. Opt. Photon. **12**, 1012 (2020).
[63] F. F. da Silva, A. Torres-Knoop, T. Coopmans, D. Maier, and S. Wehner, Quantum Sci. Technol. **6**, 035007 (2021).
[64] C. Simon, M. Afzelius, J. Appel, A. Boyer de la Giroday, S. J. Dewhurst, N. Gisin, C. Y. Hu, F. Jelezko, S. Kröll, J. H. Müller, J. Nunn, E. S. Polzik, J. G. Rarity, H. De Riedmatten, W. Rosenfeld, A. J. Shields, N. Sköld, R. M. Stevenson, R. Thew, I. A. Walmsley *et al.*, Eur. Phys. J. D **58**, 1 (2010).
[65] V. Scarani, H. Bechmann-Pasquinucci, N. J. Cerf, M. Dušek, N. Lütkenhaus, and M. Peev, Rev. Mod. Phys. **81**, 1301 (2009).
[66] G. P. Agrawal, *Fiber-Optic Communication Systems*, 2nd ed. (Wiley, New York, 1997), p. 55.
[67] S. S. Mousavi, M. Schukat, and E. Howley, in *Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016*, edited

by Y. Bi, S. Kapoor, and R. Bhatia (Springer, Cham, 2018), pp. 426–440.

[68] J. Achiam, Spinning Up in deep reinforcement learning, https://spinningup.openai.com/en/latest/ (OpenAI, 2018).

[69] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. (MIT Press, Cambridge, 2018).

[70] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, Proximal policy optimization algorithms, arXiv:1707.06347.

[71] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. M. A. Eslami, M. Riedmiller, and D. Silver, Emergence of locomotion behaviours in rich environments, arXiv:1707.02286.

[72] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, High-dimensional continuous control using generalized advantage estimation, arXiv:1506.02438.

[73] M. Minsky, Steps toward artificial intelligence Proc. IRE **49**, 8 (1961).

[74] D. P. Kingma and J. Ba, in *Proceedings of the Third International Conference on Learning Representations, San Diego, 2015*, edited by Y. Bengio and Y. LeCun (ICLR, La Jolla, 2015).

[75] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, OpenAI Gym, arXiv:1606.01540.

[76] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg *et al.*, TensorFlow: Large-scale machine learning on heterogeneous systems, 2015, software available from tensorflow.org.

[77] E. Gabriel, G. E. Fagg, G. Bosilca, T. Angskun, J. J. Dongarra, J. M. Squyres, V. Sahay, P. Kambadur, B. Barrett, A. Lumsdaine, R. H. Castain, D. J. Daniel, R. L. Graham, and T. S. Woodall, Open MPI: Goals, Concept, and Design of a Next Generation MPI Implementation, in *Proceedings, 11th European PVM/MPI Users' Group Meeting, Budapest, Hungary*, edited by D. Kranzlmüller, P. Kacsuk, J. Dongarra, Lecture Notes in Computer Science, Vol. 3241 (Springer, Berlin, Heidelberg, 2004), pp. 97–104.

[78] S. D. Reiß, Supplement to M.Sc. thesis, https://github.com/SimonReiss/Master-Thesis.

[79] A. Dawid, P. Huembeli, M. Tomza, M. Lewenstein, and A. Dauphin, Phase detection with neural networks: Interpreting the black box, New J. Phys. **22**, 115001 (2020).