Efficient Optimization of Cutoffs in Quantum Repeater Chains

Li, B.; Coopmans, T.J.; Elkouss Coronas, D.

DOI
10.1109/TQE.2021.3099003

Publication date
2021

Document Version
Final published version

Published in
IEEE Transactions on Quantum Engineering

Received April 28, 2021; revised July 5, 2021; accepted July 19, 2021; date of publication July 26, 2021;
date of current version September 2, 2021.

Digital Object Identifier 10.1109/TQE.2021.3099003

Efficient Optimization of Cutoffs in Quantum Repeater Chains

BOXI LI[1,2,3], TIM COOPMANS[2], AND DAVID ELKOUSS[2] (Member, IEEE)

[1] ETH Zürich, 8092 Zürich, Switzerland
[2] QuTech, Delft University of Technology, 2628 CJ Delft, The Netherlands
[3] Peter Grünberg Institute - Quantum Control (PGI-8), Forschungszentrum Jülich GmbH, D-52425 Jülich, Germany

Corresponding author: David Elkouss (d.elkousscoronas@tudelft.nl).

This work was supported by the QIA project (funded by European Union's Horizon 2020 under Grant 820445 and by the Netherlands Organization for Scientific Research (NWO/OCW), as part of the Quantum Software Consortium program under Project 024.003.037/3368). The work of Boxi Li was supported by the IDEA League Student Grant Programme.

ABSTRACT Quantum communication enables the implementation of tasks that are unachievable with classical resources. However, losses on the communication channel preclude the direct long-distance transmission of quantum information in many relevant scenarios. In principle, quantum repeaters allow one to overcome losses. However, realistic hardware parameters make long-distance quantum communication a challenge in practice. For instance, in many protocols an entangled pair is generated that needs to wait in quantum memory until the generation of an additional pair. During this waiting time the first pair decoheres, impacting the quality of the final entanglement produced. At the cost of a lower rate, this effect can be mitigated by imposing a cutoff condition. For instance, a maximum storage time for entanglement after which it is discarded. In this article, we optimize the cutoffs for quantum repeater chains. First, we develop an algorithm for computing the probability distribution of the waiting time and fidelity of entanglement produced by repeater chain protocols which include a cutoff. Then, we use the algorithm to optimize cutoffs in order to maximize the secret-key rate between the end nodes of the repeater chain. In this article, we find that the use of the optimal cutoff extends the parameter regime for which secret key can be generated and, moreover, significantly increases the secret-key rate for a large range of parameters.

INDEX TERMS Quantum communication, quantum repeater chains.

I. INTRODUCTION

The realization of a quantum internet [1] will allow any two parties on Earth to implement tasks that are impossible with its classical counterpart [2]. Quantum communication schemes rely on the distribution of entanglement between spatially separated parties, which in practice is precluded over long distances due to loss in the communication channel (usually glass fiber or free space). This problem can be overcome by dividing the distance between the sender and receiver of the quantum information into smaller segments, which are connected by intermediate nodes called quantum repeaters [3].

Most repeater schemes require quantum memories [4], [5]. Moreover, in many protocols an entangled pair is generated that needs to wait in a quantum memory until the generation of an additional pair. During this waiting time the first pair decoheres, reducing the quality of the final entanglement produced. At the cost of a lower rate, this effect can be mitigated by imposing a cutoff condition. For instance, a maximum storage time for entanglement after which it is discarded [6].

Cutoffs have been considered for entanglement generation in different contexts [6]–[17]. Notably, they play a key role for generating entanglement already in multipair experiments between adjacent nodes [8]. They also promise to be helpful in near-term quantum repeater experiments [9], [10], [14]. In the multirepeater case, it is possible to obtain analytical expressions for the waiting time for general families of protocols [15], [16], though in general it appears challenging to extend those methods to characterize the quality of the states produced. Santra *et al.* [11] analytically optimized the distillable entanglement for a restricted class of quantum repeater schemes.

In this article, we focus on the type of quantum repeater schemes that rely on heralded entanglement generation, purification and swapping, but not on quantum error correction codes, sometimes referred to as the "first generation of quantum repeaters" [4], [5]. In particular, we study a very general class of first-generation repeater schemes including probabilistic swapping, distillation and cutoffs and characterize their performance in the presence of memory decoherence. We sidestep the challenge of analytical characterization by computing the probability distribution of the waiting time and fidelity of the first generated entangled pair between the repeater's end nodes. For this, we improve the closed-form expressions by Brand *et al.* [18] to get faster algorithm runtimes and extend the expressions to repeater schemes which involve distillation and cutoffs. The runtime of the algorithm which evaluates these expressions is polynomial in the prespecified size of the computed probability distribution's support.

In the second part of the article, we optimize the choices of the cutoff to maximize the secret-key rate. We study different cutoff strategies and find that the use of the optimal cutoff extends the parameter regime for which secret key can be generated and moreover significantly increases the secret-key rate for a large range of parameters. We also analyze the dependence of the optimal cutoff on different properties of the hardware and find that memory quality highly influences the effectiveness of the cutoff, whereas the influence is small for success probability of entanglement swapping. In addition, our numerical simulations show that for symmetric repeater protocols with evenly spaced nodes, a nonuniform cutoff (different cutoff time in different parts of the repeater chain) does not yield a significant improvement in end-to-end node secret key rate compared to a uniform cutoff.

This article is organized as follows. In Section II, we describe the class of repeater schemes under study and elaborate on the hardware model used in our simulations. Section III presents the closed-form expressions and their evaluation algorithms for the waiting time distribution and output quantum states of repeater schemes which include cutoffs. The second part of the work, on optimization of the cutoff, consists of Section IV, where we provide details on the optimization procedure, and the results of the numerical optimization as presented in Section V. Section VI ends our work with a conclusion.

II. PRELIMINARIES

A. CLASS OF REPEATER PROTOCOLS CONSIDERED

A quantum repeater chain connects two endpoints via several repeaters and aims to generate entanglement between the endpoints. In this section, we elaborate on the class of quantum repeater chain protocols we study in this work, which is an extension of the class studied in [18] with the addition of cutoffs. While doing so, we refer to both the endpoints and the repeater stations as nodes and to an entangled state between two nodes as a link.

The class of quantum repeater protocols studied in this work are composed of the following four building blocks or PROTOCOL-UNITS: elementary link generation (GEN), entanglement swap (SWAP), entanglement distillation (DIST) and cutoff (CUTOFF). See Fig. 1(a). All of these processes can

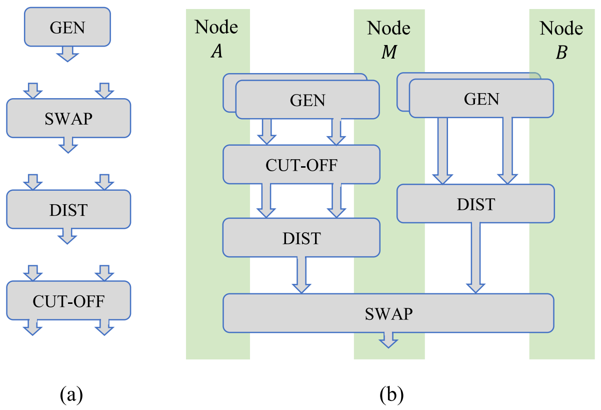

FIGURE 1. Class of repeater chain protocols considered in this work is composed of four different types of protocol-units. (a) Four protocol-units: elementary-link generation between adjacent nodes (GEN), entanglement swapping for connecting two short-distance links in a single long-distance one (SWAP), entanglement distillation for converting two low-quality links in a single high-quality link (DIST) and discarding two links (CUTOFF), for example if their generation times differ by more than a prespecified cutoff time. The repeater chain protocols we consider in this work are composed of combinations of the four PROTOCOL-UNITs, provided that each CUTOFF is succeeded by a SWAP or DIST. The in-/outgoing arrows of each protocol-unit indicate the number of entangled links that the block consumes/produces. (b) Example of a composite protocol on three nodes (end nodes *A* and *B* and single repeater *M*). At the start of the protocol, two fresh elementary links are generated (GEN) in parallel between adjacent nodes *A* and *M* and subsequently selected through a CUTOFF block. The first two links that survive the cutoff are then distilled (DIST) into a single link of higher quality. Asynchronously, the nodes *M* and *B* generate (GEN) pairs of links until the distillation (DIST) succeeds. Once distillation on both sides of node *M* has succeeded, the resulting links *A* ↔ *M* and *M* ↔ *B* are converted via a swap into a single entangled link between the end nodes *A* and *B*.

fail, but the involved nodes receive a success or failure message. That is, they are heralded. In what follows, we describe these four PROTOCOL-UNITS in more detail and subsequently explain how they can be composed into a repeater scheme that spans multiple nodes.

The first block GEN represents the generation of fresh entanglement between two adjacent nodes. We refer to those entangled pairs as an elementary link. The GEN block thus spans precisely two nodes, takes no input and outputs a single link.

The second and third blocks are entanglement swap (SWAP) and entanglement distillation (DIST). In the setting of two nodes A and B with a middle station M in between, an entanglement swap [19], [20] takes two links $A \leftrightarrow M$ and $M \leftrightarrow B$ and outputs a single link $A \leftrightarrow B$. It spans at least three nodes. Next, entanglement distillation probabilistically transforms two low-quality links between the same pair of nodes to a new one with higher quality [21], [22]. The DIST block thus spans at least two nodes, takes two links as input and outputs a single link, where each link is shared by the same pair of nodes. Both SWAP and DIST consist of local operations including measurements and classical communication. They can succeed or fail. In case of success, the nodes will keep idling until the entanglement is consumed. In case of failure, both input links are lost.

The last PROTOCOL-UNIT is CUTOFF, which takes two links as input (not necessarily between the same nodes). It accepts or rejects the two input links depending on a success condition. In case of success, it leaves the two input links untouched and outputs them again. In case of failure, both input links are discarded. In this work, we study three different success conditions. In the first two, DIF-TIME-CUTOFF and MAX-TIME-CUTOFF, "success" is declared if respectively the difference or the maximum of the input links' production times does not exceed some prespecified cutoff threshold. In the third strategy, FIDELITY-CUTOFF, the input states are passed on only if they are both of sufficient quality. This success condition translates to a cutoff on the individual input states' fidelity with a maximally entangled state (see Section II-B).

We now explain how the PROTOCOL-UNITS described above can be composed into a single repeater protocol spanning a chain of nodes. See Fig. 1(b) for an example. Each composite protocol on a chain of nodes starts with one or multiple GEN blocks between each pair of adjacent nodes for fresh elementary link generation. A protocol then consists of stacking instances of the other three PROTOCOL-UNITS in such a way that the output link(s) of one are used as input link(s) to the other. The only restriction on how the PROTOCOL-UNITS can be stacked is that both output links of CUTOFF are used as inputs for one DIST or SWAP block. As a consequence of the stacking, any repeater protocol in the class we study has a tree structure [see also Fig. 1(b)]. If a block at the root of a tree fails, then its input links are discarded and the GEN blocks at the tree's leaves will restart.

We note that the class of repeater protocols described above includes, for instance, the well-known family of repeater schemes described by Briegel *et al.* [3], [23].

B. MODEL

We here describe how we model each of the four PROTOCOL-UNITS described in Section II-A, which is identical to the modeling in [18], except for the newly introduced CUTOFF unit. For each PROTOCOL-UNIT, we describe the success condition as well as the quantum state that it outputs.

First, we model the fresh entanglement generation (GEN) using schemes which generate links in heralded attempts of duration $L_{\text{internode}}/c$, where $L_{\text{internode}}$ is the internode distance and c is the speed of light in the used transmission medium, e.g., glass fiber [4]. We assume that each attempt is independent and succeeds with constant probability $0 < p_{\text{gen}} \leq 1$. For simplicity, we assume that the nodes are equally spaced with internode distance L_0, so that each attempt in elementary link generation takes duration $\Delta t_0 = L_0/c$, which will be the time unit in our numerical simulation.

We model the elementary link as a Werner state $\rho(w)$ with constant Werner parameter $w = w_0$ [24]

$$\rho(w) = w|\Phi^+\rangle\langle\Phi^+| + (1-w)\frac{\mathbb{1}_4}{4} \tag{1}$$

where the Bell state

$$|\Phi^+\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$$

is a maximally entangled two-qubit state and

$$\mathbb{1}_4/4 = (|0\rangle\langle 0| + |1\rangle\langle 1|) \otimes (|0\rangle\langle 0| + |1\rangle\langle 1|)/4$$

is the maximally mixed state on two qubits. We refer to the parameter w with $0 \leq w \leq 1$ as the Werner parameter. Since a Werner state is completely determined by its Werner parameter, we use the Werner parameter to indicate the quantum state.

Equivalent to the Werner parameter, we will also express the state's quality using the fidelity, which for general density matrices ρ and σ is defined as

$$F(\rho,\sigma) := \text{Tr}\left(\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right)^2.$$

The fidelity between a Werner state $\rho(w)$ and $|\Phi^+\rangle\langle\Phi^+|$ equals

$$F = \frac{1+3w}{4}.$$

We emphasize that both the closed-form expressions and the algorithm we present in Section III are compatible with a more general representation of quantum states using density matrices, by replacing the Werner parameter with a density matrix. We choose the Werner state for the clarity of the presentation. It is a pessimistic approximation because any two-qubit state can be transformed into a Werner state with local operations and classical communication without changing the fidelity [25].

For the other PROTOCOL-UNITS, the success conditions are summarized in Table 1. In short: we model entanglement swapping (SWAP) as succeeding with a constant probability p_{swap}. For entanglement distillation (DIST), we use the BBPSSW protocol [21] which we adapt by bringing the output state back into Werner form. The latter operation does not change the output state's fidelity with the target state $|\Phi^+\rangle$. The success probability p_{dist} of distillation is a function of the input states' Werner parameters (see [18] for details). The cutoff (CUTOFF) success condition depends deterministically on the waiting time or the fidelity of the input links.

The states that any PROTOCOL-UNIT outputs are Werner states at any time of the execution of the protocol. Indeed, a successful entanglement swap or distillation attempt maps Werner states to Werner states (see [18] for a brief explanation). Also, the CUTOFF leaves the input states untouched in case of success, thereby outputting Werner states if it got those as input. For each PROTOCOL-UNIT, the Werner parameters of the output links w_{out} are a function of those of the input links and are given in Table 1.

In addition to the fact that the PROTOCOL-UNITS change the quantum states they handle, the local quantum memories that are used to store the links are imperfect. In our model, a link with initial Werner parameter w, which lives in memory for

TABLE 1. Overview of Success Probability and the Output Werner Parameter for Each Protocol-Unit

PROTOCOL-UNIT	success probability p	Werner parameter w_{out}
generation (GEN)	p_{gen} (constant)	w_0
entanglement swapping (SWAP)	p_{swap} (constant)	$w'_A \cdot w'_B$
entanglement distillation (DIST)	$p_{\text{dist}} = \dfrac{1 + w'_A w'_B}{2}$	$\dfrac{w'_A + w'_B + 4w'_A w'_B}{6p_{\text{dist}}}$
DIF-TIME-CUT-OFF	$p_{\text{cut}} = \begin{cases} 1 & \text{if } \lvert t_{\text{A}} - t_{\text{B}} \rvert \leq \tau \\ 0 & \text{otherwise} \end{cases}$	$w'_A,\ w'_B$
FIDELITY-CUT-OFF	$p_{\text{cut}} = \begin{cases} 1 & \text{if } w'_A \geq w_{\text{cut}} \text{ and } w'_B \geq w_{\text{cut}} \\ 0 & \text{otherwise} \end{cases}$	$w'_A,\ w'_B$
MAX-TIME-CUT-OFF	$p_{\text{cut}} = \begin{cases} 1 & \text{if} \max(t_A, t_B) \leq \tau \\ 0 & \text{otherwise} \end{cases}$	$w'_A,\ w'_B$

where $(t_{\text{A}}, w_{\text{A}})$ and $(t_{\text{B}}, w_{\text{B}})$ are the waiting time and Werner parameter of the links A and B provided as input to the PROTOCOL-UNIT. Parameters τ and w_{cut} are the cut-off thresholds on time and Werner parameter, respectively. The primed notation denotes Werner parameter with decay in (2) applied to the link that waits until the other is finished: $w'_X = w_X \cdot e^{-(\lvert t_{\text{A}} - t_{\text{B}} \rvert + t_{\text{c}})/t_{\text{coh}}}$ if X denotes the input link that finishes earlier and $w'_X = w_X \cdot e^{-t_{\text{c}}/t_{\text{coh}}}$ otherwise. The parameter t_{c} denotes the time used for classical communication and local operations. The expressions for SWAP and DIST are derived in Appendix A of Ref. [18]. For an explanation of the different PROTOCOL-UNITs, see section II-A.

time Δt until it is retrieved, decoheres to Werner parameter

$$w_{\text{decayed}} = w \cdot e^{-\Delta t / t_{\text{coh}}} \tag{2}$$

where t_{coh} is the joint coherence time of the two involved memories.

In summary, for a given composite protocol (including the cutoff condition τ or w_{cut} for each CUTOFF block), the simulation of the entanglement distribution process is determined by four hardware parameters: the success probability of elementary link generation p_{gen}, the swap success probability p_{swap}, the Werner parameter of the elementary link w_0 and the memory coherence time t_{coh}.

C. WAITING TIME AND PRODUCED END-TO-END STATE IN REPEATER SCHEMES USING PROBABILISTIC COMPONENTS

In this article, we study the time until the first entangled pair of qubits is generated between the end nodes of the repeater chain (called "waiting time" from here on) and the state's quality, expressed as its Werner parameter (recall that the end-to-end state is a Werner state, see Section II-B). Because the repeater chain protocols we study in this work are composed of probabilistic components, both the waiting time and the end-to-end state's Werner parameter are random variables. See Fig. 2 for an illustration of the random behavior of the waiting time. We characterize the quality by the averaged Werner parameters of all states generated at the same time step t. The algorithm we present in this work computes the probability distribution $\Pr(T = t)$ of the waiting time T and the average Werner parameter $W(t)$ of the end-to-end state which is delivered at time t.

We finish this section by noting that by considering the average Werner parameter, we ignore the "history" of a link, resulting in a suboptimal estimation of the fidelity of the states. To see this, consider for example the three-node protocol of Fig. 1(b). In this protocol, the following two series of events lead to an output entangled pair between nodes A and B at time $t = 10$. (i) All GEN blocks fail at each timestep $t < 10$ but succeed at time $t = 10$, after which all other PROTOCOL-UNITS also succeed immediately. (ii) The PROTOCOL-UNITS between A and M all succeed at time $t = 1$, while the GEN blocks between M and B succeed at time $t = 10$, followed by all other remaining PROTOCOL-UNITS also succeeding at time $t = 10$. In case (i), no entanglement has waited in memory, whereas in case (ii), the produced link between A and M has waited ten timesteps and decohered in that time. By keeping track of the timestamps at which the several PROTOCOL-UNITS succeeded, one could distinguish these two scenarios. Since the resulting fidelity estimation computation is rather complex and in this work, we focus on quantifying the effect of a cutoff, we leave such advanced fidelity estimation for future work.

III. COMPUTING THE WAITING TIME DISTRIBUTION AND THE OUTPUT WERNER PARAMETER

In this section, we present closed-form expressions of the waiting time probability distribution and Werner parameter of the output links for each PROTOCOL-UNIT, as function of waiting time distribution and Werner parameter of its input links. Expressions for a composite protocol are obtained by iterative application over the PROTOCOL-UNITS that the protocol consists of. These expressions naturally lead to an algorithm for their evaluation, which we also present in this section.

Closed-form expressions for GEN and SWAP were already obtained by Brand *et al.* [18], who explicitly mentioned that their approach does not generalize straightforwardly to DIST. Here, we include DIST and even CUTOFF, provided

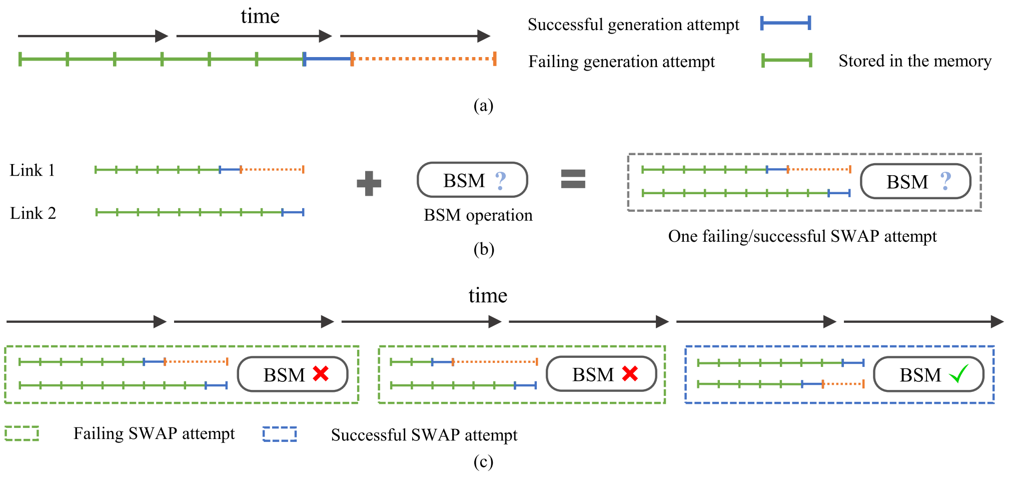

FIGURE 2. Visualization of the waiting time until end-to-end entanglement is delivered for a three-node repeater chain. The repeater scheme consists of the generation of two elementary links, followed by an entanglement swap on the two links. (a) Single link is generated in fixed-duration attempts, which succeed probabilistically and thus may fail (green line segment), after which generation is re-attempted until success (blue line segment). After that, the link is stored until it is consumed (dotted orange line segment). (b) Run of the three-node protocol until the first swap attempt, which consists of first preparing two input links in parallel, followed by a Bell state measurement (BSM). The link that is generated earlier than the other needs to wait in the memory (link 1 in the figure, the "waiting" is indicated by the dotted orange line). While waiting, the earlier link's quality decreases due to decoherence. The total waiting time before the BSM equals the maximum of the generation times of the two links. The BSM operation can fail, in which case the two links are lost and need to be regenerated. (c) Full run of the three-node protocol, consisting of failed entanglement swaps (green dashed box) on fresh links until the first successful swap (blue dashed box). The total waiting time is the sum of the waiting times for the parallel generation of each pair of elementary links, up to and including the first successful swap.

the latter is succeeded by SWAP or DIST. The novel idea is to use separate expressions for the waiting time probability distribution of a successful and failed attempt. We then express the total waiting time distribution and the Werner parameter as those of the successful attempt averaged by the occurrence probability of all possible sequences of failed attempts, where the weighted average is efficiently computed using convolution. As an additional benefit, the evaluation algorithm for SWAP is faster than the one presented by Brand *et al.*

In the following, we first derive general closed-form expressions for the waiting time distribution and Werner parameter of one PROTOCOL-UNIT in Section III-A. We then give specific expressions for each PROTOCOL-UNIT individually in Sections III-B to III-E. In the last section (Section III-F), we show how these expressions can be converted into an efficient algorithm. We also explain how to modify the closed-form expressions using the discrete Fourier transform, motivated by its use in [26] and [27]. These modified expressions lead to an even faster algorithm for computing the waiting time and Werner parameter, which we provide in Appendix B. We denote the random variables of the waiting time and average Werner parameter as T and $W(t)$, with subscript A and B for the input links and "out" for the output link (see Fig. 3).

A. GENERAL CLOSED-FORM EXPRESSIONS FOR WAITING TIME AND PRODUCED STATES FOR ALL PROTOCOL-UNITS

1) RANDOM VARIABLE EXPRESSION FOR THE WAITING TIME OF PROTOCOL-UNITS

We start by presenting an expression for the random variable T_{out}. To study the waiting time distribution, we divide the total waiting time into the waiting time for each attempt. An attempt can fail or succeed and it repeats until the first successful attempt occurs (see Fig. 2). The total waiting time T_{out} is given by

$$T_{\text{out}} = \sum_{i=1}^{K} M^{(i)} \tag{3}$$

where $M^{(i)}$ are i.i.d. random variables characterizing the waiting time of each attempt and therefore each is a function of the waiting time of two input links T_{A}, T_{B}. For example, for SWAP, we have $M = \max(T_{\text{A}}, T_{\text{B}})$, i.e., we need to wait until both links are ready to perform the operation. K is the number of attempts we need until the first successful attempt occurs, which is also a random variable.

Since the success probability of one attempt can be time-dependent, in general, K is correlated to $M^{(j)}$. To make this correlation between K and M in (3) explicit, we introduce a random variable Y. Y denotes the binary random variable describing success (1) or failure (0) of a single attempt, subjected to the success probability p. The success probability p

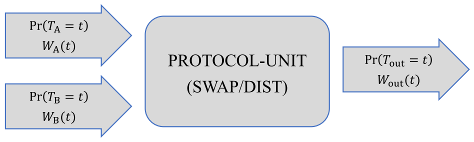

FIGURE 3. Workflow of the algorithm for one protocol-unit (swap or dist). It takes the waiting time distribution and Werner parameter of the two input links and computes those of the output.

of one attempt is given by $p = p(t_A, t_B)$ in Table 1 and it can be understood as the success probability with given waiting time t_A, t_B of the input links

$$p(t_A, t_B) = \Pr(Y = 1 | T_A = t_A, T_B = t_B).$$

Notice that, although the success probability can depend on Werner parameters, i.e., $p(t_A, t_B, w_A, w_B)$, we can always reduce the Werner parameter dependence to time dependence by plugging in $w_A = W_A(t_A)$ and $w_B = W_B(t_B)$.

With the above definition, we rewrite (3) with a sum over all possible number of attempts weighted by its occurrence probability [6]

$$T_{\text{out}} = \sum_{k=1}^{\infty} \left\{ \left(Y^{(k)} \prod_{j=1}^{k-1} \left(1 - Y^{(j)}\right) \right) \cdot \sum_{i=1}^{k} M^{(i)} \right\}. \tag{4}$$

The expression in round brackets evaluates to 1 precisely if $Y^{(k)} = 1$ and $Y^{(j)} = 0$ for all $j < k$, and to 0 in all other cases. This factor thus makes that only the sum $\sum_{i=1}^{k} M^{(i)}$ is taken for which k is the first successful attempt. Notice that $Y^{(j)}$ and $M^{(i)}$ are correlated for all $i = j$ because they describe the same attempts. In the next section, we go further to compute the probability distribution of T_{out}.

2) CLOSED-FORM EXPRESSION FOR THE WAITING TIME DISTRIBUTION

In the following, we give an expression of the waiting time distribution $\Pr(T_{\text{out}} = t)$ for one PROTOCOL-UNIT.

We consider the generation time of a successful or failed attempt separately and use the joint distribution of M and Y. We define the joint distribution that one attempt succeeds/fails and takes time t as

$$\begin{aligned} P_s(t) &:= \Pr(M = t, Y = 1) \\ &= \sum_{t_A, t_B : \max(t_A, t_B) + t_c = t} \Pr(T_A = t_A, T_B = t_B) \cdot p(t_A, t_B) \end{aligned} \tag{5}$$

$$\begin{aligned} P_f(t) &:= \Pr(M = t, Y = 0) \\ &= \sum_{t_A, t_B : \max(t_A, t_B) + t_c = t} \Pr(T_A = t_A, T_B = t_B) \cdot [1 - p](t_A, t_B) \end{aligned} \tag{6}$$

where t_c is the time used for classical communication and local operations. In the above equation, we iterate over all possible combinations of the input links' generation time t_A, t_B that leads to a waiting time t for this attempt.

With the definition (5) and (6), the sum of the waiting time for all attempts can be obtained by

$$\Pr(T_{\text{out}} = t) = \sum_{k=1}^{\infty} \left[\left(\mathop{*}_{j=1}^{k-1} P_f^{(j)} \right) * P_s \right](t) \tag{7}$$

where $*$ is the notation for convolution and the sum over k considers all the possible numbers of attempts. The notation $*_{j=1}^{k-1} P_f^{(j)}$ represents the convolution of $k-1$ independent functions P_f. In the above equation, the discrete linear convolution is defined by

$$[f_1 * f_2](t) = \sum_{t'=0}^{t} f_1(t - t') \cdot f_2(t'). \tag{8}$$

If f_1, f_2 describe two probability distributions of two random variables, their convolution is the distribution of the sum of those two random variables. However, neither P_f or P_s characterizes a random variable since they are joint distributions including Y. That is to say, P_s and P_f do not sum up to 1. Instead, we have

$$\sum_t P_f(t) + \sum_t P_s(t) = 1. \tag{9}$$

Therefore, the convolution here cannot be simply interpreted as a sum of two random variables. Instead, it is the summed waiting time conditioned on the success/failure of each attempt.

As we will show in Section III-F, (7) is sufficient for the derivation of the main algorithm for computing the probability distribution of T_{out} we present in this work. The algorithm's runtime is partially determined by the sum and the convolution in the summand in (7). Fortunately, these can be eliminated by the use of the discrete Fourier transform, resulting in a faster alternative algorithm. Below, we use the Fourier transform to derive an equivalent expression to (7). The alternative algorithm is given in Appendix B.

Since the discrete Fourier transform acts on a finite sequence of numbers, we first truncate the probability distribution at a fixed time L, i.e., we obtain the finite sequence $\{\Pr(T_{\text{out}} = t) | t = 0, 1, 2, \ldots, L\}$. If $\vec{x} := x_0, x_1, \ldots, x_{L-1}$ is a sequence of complex numbers, then its Fourier transform $\mathcal{F}(\vec{x})$ is the sequence $y_0, y_1, \ldots, y_{L-1}$ given by

$$y_j = \sum_{k=0}^{L-1} x_k \cdot \exp(-2\pi \mathrm{i} \cdot j \cdot k / L) \tag{10}$$

where i is the complex unit. The Fourier transform is a linear map and moreover it converts convolutions into element-wise multiplication, i.e., $\mathcal{F}(\vec{x} * \vec{x}') = \mathcal{F}(\vec{x}) \cdot \mathcal{F}(\vec{x}')$. As a consequence, taking the Fourier transform of both sides of (7) yields

$$\mathcal{F}[\Pr(T_{\text{out}} = t)] = \sum_{k=1}^{\infty} \left[\left(\prod_{j=1}^{k-1} \mathcal{F}\left(P_f^{(j)}\right) \right) \cdot \mathcal{F}[P_s] \right](t).$$

Because $P_f^{(j)}$ are identical distribution for all j, we use the identity $\sum_{k=1}^{\infty} x^{(k-1)} = 1/(1-x)$ to obtain

$$\Pr(T_{\text{out}} = t) = \mathcal{F}^{-1} \left[\frac{\mathcal{F}[P_s]}{1 - \mathcal{F}[P_f]} \right](t). \tag{11}$$

3) CLOSED-FORM EXPRESSION FOR THE WERNER PARAMETER

Here, we derive the expression for the Werner parameter $W_{\text{out}}(t)$.

To arrive at $W_{\text{out}}(t)$, we first compute the average Werner parameter of the output link of one attempt, given that it succeeds and finishes at time t

$$W_{\text{s}}(t) = \frac{\sum_{t_{\text{A}},t_{\text{B}}:\max(t_{\text{A}},t_{\text{B}})+t_{\text{c}}=t} \Pr(T_{\text{A}}=t_{\text{A}}, T_{\text{B}}=t_{\text{B}})\cdot[p\cdot w_{\text{out}}](t_{\text{A}},t_{\text{B}})}{P_{\text{s}}(t)}. \tag{12}$$

Here, w_{out} is the Werner parameter of the output link of a successful attempt and p the success probability (see Table 1). We again simplify the notation with $w_{\text{out}}(t_{\text{A}},t_{\text{B}}) = w_{\text{out}}(t_{\text{A}},t_{\text{B}},W_{\text{A}}(t_{\text{A}}),W_{\text{B}}(t_{\text{B}}))$.

Next, we take a weighted average of W_{s}' over all possible sequences of failed attempts, followed by a single successful attempt:

$$W_{\text{out}}(t) = \frac{\sum_{k=1}^{\infty}\left[\left(\mathop{*}_{j=1}^{k-1} P_{\text{f}}^{(j)}\right) * (P_{\text{s}}\cdot W_{\text{s}})\right](t)}{\Pr(T_{\text{out}}=t)} \tag{13}$$

where $*_{j=1}^{k-1} P_{\text{f}}^{(j)}$ computes the waiting time distribution of $k-1$ failed attempts and the additional convolution is the weighted average.

For (7), which is an expression for the probability distribution of T_{out}, we obtained a more compact equivalent, (11), by moving to Fourier space. By an analogous derivation, we can get a more compact expression for W_{out} than (13)

$$W_{\text{out}}(t) = \mathcal{F}^{-1}\left[\frac{\mathcal{F}[P_{\text{s}}\cdot W_{\text{s}}]}{1-\mathcal{F}[P_{\text{f}}]}\right]\frac{1}{\Pr(T_{\text{out}}=t)}(t). \tag{14}$$

B. SPECIFIC CASE: GEN

We give here the expression for PROTOCOL-UNIT GEN. Since GEN does not have input links, the output does not rely on the expression introduced in the Section III-A. Because one attempt in GEN takes one time step and the success probability p_{gen} is a constant, the waiting time can be described by a geometric distribution

$$\Pr(T_{\text{out}}=t) = p_{\text{gen}}(1-p_{\text{gen}})^{t-1}.$$

The output state is a Werner state with Werner parameter w_0 as described in Section II-B.

C. SPECIFIC CASE: SWAP

For entanglement swap, since p_{swap} is constant, Y is not correlated with M. As a result, P_{s} and P_{f} differ only by a constant coefficient [see (5) and (6)]. Therefore, we can factor the constant out and get

$$\Pr(T_{\text{out}}=t) = \sum_{k=1}^{\infty} p_{\text{swap}}(1-p_{\text{swap}})^{k-1}\left[\mathop{*}_{j=1}^{k} m\right] \tag{15}$$

where

$$m(t) := Pr(M=t) = \sum_{t_{\text{A}},t_{\text{B}}:\max(t_{\text{A}},t_{\text{B}})=t} \Pr(T_{\text{A}}=t_{\text{A}},T_{\text{B}}=t_{\text{B}}).$$

The constant time t_{c} is set to 0. This is exactly the geometric compound distribution obtained in [18].

For the Werner parameter, we can directly use (13) and obtain

$$W_{\text{out}} = \sum_{k=1}^{\infty} p_{\text{swap}}(1-p_{\text{swap}})^{k-1}\left[\left(\mathop{*}_{j=1}^{k-1} m\right) * \left(m\cdot W_{\text{s}}'\right)\right]. \tag{16}$$

Compared to the expression in [18], this expression replaces the iteration over all pair of possible input Werner parameters for each k by convolution.

Both expressions above can also be written in Fourier space by substituting $P_{\text{s}} = p_{\text{swap}}m(t)$ and $P_{\text{f}} = (1-p_{\text{swap}})m(t)$ in (11) and (14).

D. SPECIFIC CASE: DIST

For entanglement distillation, the success probability depends on the Werner parameters. As discussed in Section III-A, we can compute T_{out} and W_{out} because we iterate over all possible combinations of t_{A} and t_{B} and we use $W(t)$ to reduce the dependence on Werner parameters to the dependence on the waiting time. The calculation goes as follows. First, we compute P_{f} and P_{s} using $p(t_A,t_B) = p_{\text{dist}}(W(t_A),W(t_B))$ (see Table 1). Then, we plug in P_{f} and P_{s} in (7) to compute T_{out}. Finally, W_{out} can be calculated similarly using Table 1, (12), and (13).

E. SPECIFIC CASE: CUT-OFF

CUTOFF selects the input links and accepts them if the cutoff condition described in Section II-B is fulfilled. We consider only the case where CUTOFF is followed by SWAP or DIST, so that the two blocks together output a single entangled link.

1) WAITING TIME DISTRIBUTION

We define a new binary variable Y_{cut} representing whether the cutoff condition is fulfilled. The corresponding success probability is described by p_{cut} in Table 1. In addition, we also define the waiting time of one cutoff attempt as Z, in contrast to M for a swap or distillation attempt. For CUTOFF, we need to distinguish the waiting time of a successful and a failed attempt. In the case of success, we always have $Z_{\text{s}} = \max(T_{\text{A}},T_{\text{B}})$, i.e., we wait until two links are produced. However, in the case of failure, the waiting time is different for different cutoff strategies. With the notation $Z_{\text{f}} = t_{\text{fail}}(T_{\text{A}},T_{\text{B}})$, we have the following: for DIF-TIME-CUTOFF, $t_{\text{fail}}(T_{\text{A}},T_{\text{B}}) = \min(T_{\text{A}},T_{\text{B}})+\tau$, because there is no need to wait for the second link longer than the cutoff threshold. For MAX-TIME-CUTOFF, $t_{\text{fail}}(T_{\text{A}},T_{\text{B}})$ is the constant τ, i.e., the maximal allowed waiting time. For FIDELITY-CUTOFF, it is $t_{\text{fail}}(T_{\text{A}},T_{\text{B}}) = \max(T_{\text{A}},T_{\text{B}})$.

Similar as the nested structure shown in Fig. 2, a swap or distillation attempt is now composed of several cutoff attempts. We can write its waiting time M as

$$M = \sum_k \left\{ \left[Y_{\mathrm{cut}}^{(k)} \prod_{j=1}^{k-1} \left(1 - Y_{\mathrm{cut}}^{(j)}\right) \right] \cdot \left[Z_{\mathrm{s}}^{(k)} + \sum_{i=1}^{k-1} \left(Z_{\mathrm{f}}^{(i)} \right) \right] \right\}.$$

This expression will replace $M = \max(T_{\mathrm{A}}, T_{\mathrm{B}})$ used in (4). For $\tau = \infty$ or $w_{\mathrm{cut}} = 0$, i.e., no cutoff, Y_{cut} is always 1. Therefore, $k = 1$ is the only surviving term and the two expressions coincide.

To calculate the waiting time distribution, we need three joint distributions—P'_{f} for unsuccessful input link preparation because of the cutoff, $P'_{\mathrm{s,f}}$ for successful preparation but unsuccessful swap/distillation, and $P'_{\mathrm{s,s}}$ for both successful

$$\begin{aligned}
P'_{\mathrm{f}}(t) &= \Pr(M = t, Y_{\mathrm{cut}} = 0) \\
&= \sum_{t_{\mathrm{A}}, t_{\mathrm{B}} :\, t_{\mathrm{fail}}(t_{\mathrm{A}}, t_{\mathrm{B}}) + t_{\mathrm{c}} = t} \Pr(T_{\mathrm{A}} = t_{\mathrm{A}}, T_{\mathrm{B}} = t_{\mathrm{B}}) \\
&\quad \cdot [1 - p_{\mathrm{cut}}](T_{\mathrm{A}}, T_{\mathrm{B}}) \\
P'_{\mathrm{s,f}}(t) &= \Pr(M = t, Y_{\mathrm{cut}} = 1, Y = 0) \\
&= \sum_{t_{\mathrm{A}}, t_{\mathrm{B}} :\, \max(t_{\mathrm{A}}, t_{\mathrm{B}}) + t_{\mathrm{c}} = t} \Pr(T_{\mathrm{A}} = t_{\mathrm{A}}, T_{\mathrm{B}} = t_{\mathrm{B}}) \\
&\quad \cdot [p_{\mathrm{cut}} \cdot (1 - p)](t_{\mathrm{A}}, t_{\mathrm{B}}) \\
P'_{\mathrm{s,s}}(t) &= \Pr(M = t, Y_{\mathrm{cut}} = 1, Y = 1) \\
&= \sum_{t_{\mathrm{A}}, t_{\mathrm{B}} :\, \max(t_{\mathrm{A}}, t_{\mathrm{B}}) + t_{\mathrm{c}} = t} \Pr(T_{\mathrm{A}} = t_{\mathrm{A}}, T_{\mathrm{B}} = t_{\mathrm{B}}) \\
&\quad \cdot [p_{\mathrm{cut}} \cdot p](t_{\mathrm{A}}, t_{\mathrm{B}}).
\end{aligned}$$

The prime notation indicates that they describe the waiting time of one attempt in CUTOFF, in contrast to one attempt in swap or distillation.

For one attempt in swap/distillation with time-out, we then get similarly to (7)

$$P_{\mathrm{s}}(t) = \Pr(M = t, Y = 1) = \sum_k \left[\left(\mathop{\ast}_{j=1}^{k-1} P_{\mathrm{f}}^{\prime(j)} \right) \ast P'_{\mathrm{s,s}} \right](t)$$

$$P_{\mathrm{f}}(t) = \Pr(M = t, Y = 0) = \sum_k \left[\left(\mathop{\ast}_{j=1}^{k-1} P_{\mathrm{f}}^{\prime(j)} \right) \ast P'_{\mathrm{s,f}} \right](t)$$

as well as the expressions in Fourier space analogous to (11)

$$P_{\mathrm{s}}(t) = \Pr(M = t, Y = 1) = \mathcal{F}^{-1}\left[\frac{\mathcal{F}[P'_{\mathrm{s,s}}]}{1 - \mathcal{F}[P'_{\mathrm{f}}]} \right]$$

$$P_{\mathrm{f}}(t) = \Pr(M = t, Y = 0) = \mathcal{F}^{-1}\left[\frac{\mathcal{F}[P'_{\mathrm{s,f}}]}{1 - \mathcal{F}[P'_{\mathrm{f}}]} \right].$$

The total waiting time then follows by substituting the expressions for P_{f} and P_{s} above in (7) or (11).

For entanglement swap, i.e., constant success probability p_{swap}, simplification can be made for this calculation. In this special case, $P'_{\mathrm{s,f}}$ and $P'_{\mathrm{s,s}}$ differ only by a constant and the same holds for P_{s} and P_{f}.

2) WERNER PARAMETER

For the Werner parameter, we now need three steps.

We start from calculating the resulting Werner parameter of a swap or distillation for the very last preparation attempt where $Y_{\mathrm{cut}} = Y = 1$. It is denoted by W'_{s} and we only need to replace P_{s} by $P'_{\mathrm{s,s}}$ and $p \cdot w_{\mathrm{out}}$ by $p_{\mathrm{cut}} \cdot p \cdot w_{\mathrm{out}}$ in (12).

Next, we compute the Werner parameter $W_{\mathrm{s}}(t)$ as a function of time t that includes the failed cutoff attempts, in analog to the derivation of (13). $W_{\mathrm{s}}(t)$ is the Werner parameter that the pair of output links of CUTOFF will produce, given that the swap or distillation operation following is successful:

$$W_{\mathrm{s}}(t) = \frac{\sum_{k=1}^{\infty} \left[\left(\mathop{\ast}_{j=1}^{k-1} P'_{\mathrm{f}} \right) \ast (P'_{\mathrm{s,s}} \cdot W'_{\mathrm{s}}) \right](t)}{P_{\mathrm{s}}(t)}.$$

Finally, we consider the time consumed by failed attempts in SWAP or DIST and obtain

$$W_{\mathrm{out}}(t) = \frac{\sum_{k=1}^{\infty} \left[\left(\mathop{\ast}_{j=1}^{k-1} P_{\mathrm{f}} \right) \ast (P_{\mathrm{s}} \cdot W_{\mathrm{s}}) \right](t)}{\Pr(T_{\mathrm{out}} = t)}.$$

Using the Fourier transform, the two expressions above become

$$W_{\mathrm{s}}(t) = \mathcal{F}^{-1}\left[\frac{\mathcal{F}[P'_{\mathrm{s,s}} \cdot W'_{\mathrm{s}}]}{1 - \mathcal{F}[P'_{\mathrm{f}}]} \right] \frac{1}{P_{\mathrm{s}}}$$

$$W_{\mathrm{out}}(t) = \mathcal{F}^{-1}\left[\frac{\mathcal{F}[P_{\mathrm{s}} \cdot W_{\mathrm{s}}]}{1 - \mathcal{F}[P_{\mathrm{f}}]} \right] \frac{1}{\Pr(T_{\mathrm{out}} = t)}.$$

F. CONVERTING THE CLOSED-FORM EXPRESSIONS INTO AN EFFICIENT ALGORITHM

In the sections above, we presented closed-form expressions for T_{out} and W_{out} for each of the four PROTOCOL-UNITS, as a function of waiting time distribution and Werner parameter of the input links. In order to convert these expressions into an algorithm, we take the same approach as in [18] and cap the infinite sum in (7) and (13) by a prespecified truncation time t_{trunc}. This yields a correct $\Pr(T_{\mathrm{out}} = t)$ and $W_{\mathrm{out}}(t)$ for $t \in \{1, \ldots, t_{\mathrm{trunc}}\}$ since in each of the expressions with an infinite sum above, $\Pr(T_{\mathrm{out}} = t)$ and $W_{\mathrm{out}}(t)$ are only dependent on waiting time and Werner parameter of input links produced at time $t' \le t$.

We now show that the algorithm scales polynomially in terms of t_{trunc}. To analyze the complexity, we divide the algorithm into two parts: computing the distribution for one attempt, i.e., the iteration over all possible values of T_{A}, T_{B} [(5), (6), and (12)] and for the whole PROTOCOL-UNIT[(7) and (13)].

The complexity for the first part is $\mathcal{O}(t_{\mathrm{trunc}}^2)$ since it iterates over two discrete random variables up to t_{trunc}. For the second part, because we need at least one time step in each attempt, i.e., $\Pr(T = 0) = 0$, only the first t_{trunc} convolutions will have nonzero contribution. We can perform the convolution iteratively for each k using at most t_{trunc} convolutions. The complexity of one convolution with fast Fourier transform (FFT) is $\mathcal{O}(t_{\mathrm{trunc}} \log t_{\mathrm{trunc}})$ [28]. Thus, the complexity of the second part scales as $\mathcal{O}(t_{\text{trunc}}^2 \log t_{\text{trunc}})$. The overall complexity, therefore, is $\mathcal{O}(t_{\text{trunc}}^2 \log t_{\text{trunc}})$.

In Appendix B, we show that with further simplification of (5) and (6) as well as expressions in Fourier space [see (11) and (14)], the complexity can be reduced to $\mathcal{O}(t_{\text{trunc}} \log t_{\text{trunc}})$, with an exponentially vanishing error.

The preceding discussion shows that the algorithm is efficient as a function of the truncation time. However, for fixed truncation time, the probability mass captured by the algorithm decreases as the number of nodes increases. For protocols without cutoff, variations of the arguments in [18] would allow to prove that the algorithm introduced here is also efficient for fixed probability mass. Unfortunately, the arguments do not translate to protocols with cutoff. This is because for these protocols, the truncation time that covers a fixed probability mass can grow exponentially with the number of nodes, i.e., such an algorithm can not exist.

As an example, consider a nested protocol on 2^n repeater segments ($n = 0, 1, 2, \ldots$), which for $n = 1$ consists of a GEN block only, and for each additional level $n > 1$, each pair of adjacent links is connected by a CUTOFF followed by a SWAP. We set $\tau = 0$ for each cutoff, i.e., all elementary links need to be generated at the same time and also all entanglement swaps should succeed at the first attempt for the links to survive all the cutoffs. Since 2^n elementary links need to be generated and the protocol consists of $2^n - 1$ swaps, the probability of successful end-to-end entanglement before time t equals $1 - (1-p)^t$ with $p = p_{\text{gen}}^{N-1} \cdot p_{\text{swap}}^{N-2}$, i.e., decreases exponentially in the number of nodes $N = 2^n + 1$.

IV. OPTIMIZATION

In this section, we describe the details of our optimization over cutoffs, including the figure of merit and optimization method.

In our numerical study, we use the secret-key rate of the BB84 protocol [29] as a figure of merit to assess the performance of composite repeater protocols. We compute the secret-key rate R as the secret-key fraction divided by the average waiting time

$$R = \frac{r}{\bar{T}}. \tag{17}$$

The secret-key fraction r describes the amount of secret key that can be extracted from the generated entanglement and is given by [30] and [31]

$$r(w) = \max\{0, 1 - h[e_X(w)] - h[e_Z(w)]\} \tag{18}$$

where $h(p) = -p\log_2(p) - (1-p)\log_2(1-p)$ is the binary entropy function and e_X (e_Z) is the quantum bit error rate in the X (Z) basis. Since the quantum states tracked by our algorithm are Werner states at any point in the execution of the composite repeater protocol (see Section II), the quantum bit error rate can be expressed as function of the end-to-end state's Werner parameter

$$e_Z(w) = \langle 01|\rho(w)|01\rangle + \langle 10|\rho(w)|10\rangle = \frac{1-w}{2}$$

for a Werner state $\rho(w)$ defined in (1). The same result holds for e_X because of the symmetry of the Werner state. In Appendix C, we detail how we compute the secret-key rate with truncated waiting time distribution and Werner parameter obtained from the algorithm in Section III-F.

Since we have discrete time steps, we need an optimization algorithm which is compatible with a discrete search space. We choose the differential evolution algorithm implemented in the SciPy-optimization library of the Python programming language [32], [33].

V. NUMERICAL RESULTS

In this section, we optimize over repeater protocols with cutoffs in order to maximize the rate at which secret key can be extracted from the produced end-to-end entanglement. First, we use our algorithm from Section III and the DIF-TIME-CUTOFF strategy (see Section II) to study the effect of the cut-off on the waiting time and fidelity and show that the use of a cutoff boosts secret-key rate. We then extend our study to two other cutoff strategies, MAX-TIME-CUTOFF and FIDELITY-CUTOFF, and compare their performance. For all three cutoff strategies, we observe that the resulting repeater protocols produce secret key at significantly higher rates than their no-cutoff alternatives. Finally, we focus on the DIF-TIME-CUTOFF strategy and analyze the sensitivity of the optimal cutoff threshold with respect to the hardware parameters.

We investigate repeater protocols with three nesting levels where at each nesting level the range of entanglement is doubled by an entanglement swap. The protocol thus spans $2^3 = 8$ segments ($8 + 1 = 9$ nodes). Each entanglement swap operation is preceded by a cutoff, i.e., the scheme is of the form

$$\text{GEN} \rightarrow (\rightarrow \text{CUTOFF} \rightarrow \text{SWAP})^3. \tag{19}$$

Because cutoff is aimed to mitigate the loss of decay during the storage, we focus on parameter regimes where memory decoherence is the bottleneck for achieving high-fidelity entanglement.

As shown in (19), we consider in the numerical study only protocols without distillation, which allows us to study the improvement that cutoff brings to repeater protocols. We leave a systematic study of distillation protocols with cutoff for future work. Also, for simplicity, we ignore the time needed for classical communication for SWAP as well as the time to perform the local operations. The difference is negligible since in the parameter regimes studied, most of the time is used in repeating failed attempts rather than in communication, as investigated in [18, Fig. 6].

The numerical results in this section were obtained using our open-source implementation [34] of the algorithm from Section III on consumer-market hardware (Intel i7-8700 CPU). We validated correctness of the implementation by comparison with an extended version of the Monte Carlo algorithm from [18] (see Fig. 4 and Appendix B for details).

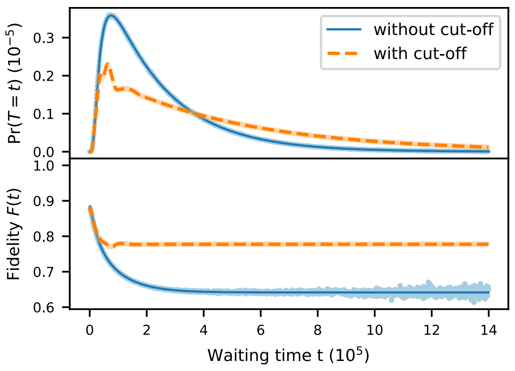

FIGURE 4. **Probability distribution of the waiting time T and the average fidelity $F(t)$ of the end-to-end link for a protocol with and without a cutoff on entanglements' production time differences (solid and dashed lines) for a nine-node repeater protocol of the form as in (19) (unit of time is the attempt duration of elementary link generation, L_0/c). We observe that the fidelity increases for most times t while the probability that the link is produced at time t shifts to larger t, indicating a longer waiting time. The secret-key rates computed from the data are 0 (without cutoff) and $0.32 \cdot 10^{-7}$ (with cutoff). The parameters used are $p_{\text{gen}} = 10^{-4}$, $p_{\text{swap}} = 0.5$, $w_0 = 0.98$, $t_{\text{coh}} = 4 \cdot 10^5$ and the cutoffs for the three nesting levels are $\tau = (1.7, 3.2, 5.5) \cdot 10^4$ (in increasing order of number of segments spanned by the cutoff block). Computation time $\approx$ 20 s for $3 \cdot 10^6$ time steps. We observe good agreement with a Monte Carlo algorithm (dots), which we use for validating the correctness of our implementation (see Appendix A for details).**

A. EFFECT OF DIF-TIME-CUT-OFF ON THE WAITING TIME AND FIDELITY

We start by investigating the DIF-TIME-CUTOFF strategy, where links are discarded if their production times differ by more than a predetermined threshold τ. We compute waiting time and average fidelity for a particular choice of the cutoff threshold at each of the three levels and compare it with the protocol without cutoff (cutoff duration $\tau = \infty$ at each nesting level), see Fig. 4. We observe that the cut-off increases fidelity at the cost of longer waiting time, as one would intuitively expect. We further quantify the time-fidelity tradeoff for a range of cutoffs in Fig. 5. For maximizing the secret key rate, we observe a single optimal choice of the cutoff threshold τ.

B. EXTENSION TO OTHER CUTOFF STRATEGIES

We extend the analysis of the previous section to two other cutoff strategies: a cutoff on the fidelity (FIDELITY-CUTOFF) and on the total waiting time (MAX-TIME-CUTOFF, see Section II and Table 1 for definitions). To be precise, we choose the same nine-node protocol from (19) and use FIDELITY-CUTOFF and MAX-TIME-CUTOFF as the CUTOFF unit, respectively.

We observe that a single optimal cutoff threshold exists for both strategies, as we saw before already for the DIF-TIME-CUTOFF strategy in Fig. 5. For each strategy, we optimize their cutoff parameters and plot the waiting time distribution and fidelity distribution in Fig. 6. As shown in the figure, although the FIDELITY-CUTOFF yields the highest secret-key rate, the distribution and resulting secret-key rate of the

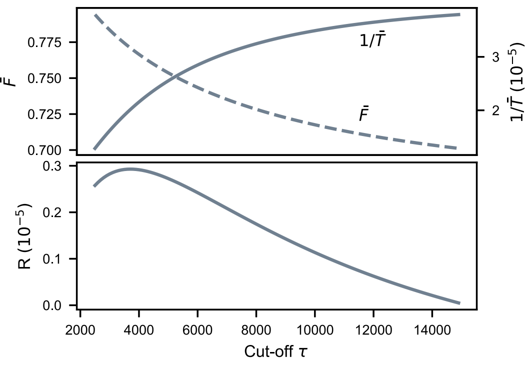

FIGURE 5. **Influence of choice of cutoff on average waiting time, average fidelity and secret-key rate for repeater protocols of the form (19) where the cutoff strategy is DIF-TIME-CUTOFF. (Top) Increasing the cutoff yields higher average generation rate (reciprocal of average waiting time $\bar{T}$) but lower average fidelity $\bar{F}$. (Bottom) The secret key rate R as a function of the cutoff time. The used parameters are $p_{\text{gen}} = 10^{-3}$, $p_{\text{swap}} = 0.5$, $w_0 = 0.98$ and $t_{\text{coh}} = 4 \cdot 10^4$. The chosen truncation time is $5 \cdot 10^5$. The cutoff time is chosen identical for all three swap levels. Unit of time is the attempt duration of elementary link generation.**

DIF-TIME-CUTOFF strategy are very close to those of the FIDELITY-CUTOFF strategy. In contrast, the MAX-TIME-CUTOFF strategy performs comparably worse in the achieved secret-key rate ($\approx 10\%$). We find similar behavior also in other parameter regimes.

Since the DIF-TIME-CUTOFF strategy is straightforward to implement in experiments while it performs only marginally worse than the best of the three strategies (FIDELITY-CUTOFF), we focus on this strategy for further analysis.

C. PERFORMANCE OF THE OPTIMAL CUTOFF FOR VARYING HARDWARE PARAMETERS

We proceed with optimizing the cutoff in the DIF-TIME-CUTOFF strategy to maximize the secret key rate for a range of parameters. The maximal secret-key rates for different repeater parameters are shown in Fig. 7(a)–(d). We observe that cutoffs extend the parameter regime for which secret key can be generated. One can also read from the figures how much cutoff lowers the requirements on hardware parameters for a given target secret key rate. To see how much one can gain in the secret key rate by using cutoffs, we choose two parameters t_{coh} and w_0 and plot the absolute increase in Fig. 8. We observe that the use of the optimal cutoff increases the secret key rate for the entire parameter range plotted and the improvement is largest close to the threshold parameters at which the no-cutoff protocol starts to produce nonzero secret key.

In addition, we compare uniform and nonuniform cutoffs, where "uniform" means that we choose the same cutoff time for each nesting level. For the parameter regimes studied, we observe that nonuniform and uniform cutoff perform similarly, see Fig. 7(a)–(d).

Our next step is the sensitivity analysis of cutoff performance in the hardware parameters. For this, we first choose baseline values for the four hardware parameters and find the corresponding optimal cutoff τ_{baseline}. Given a target set

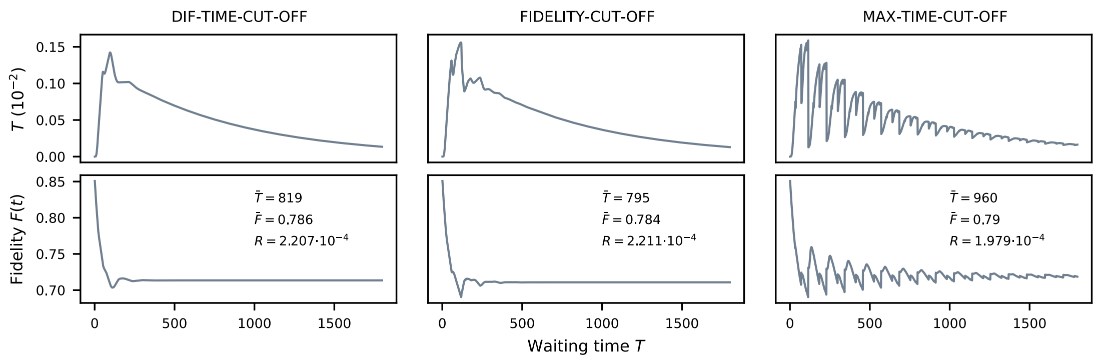

FIGURE 6. Comparison between three different cutoff strategies: cutoff on the difference of entanglements' production time (DIF-TIME-CUTOFF), the fidelity (FIDELITY-CUTOFF) and the total waiting time (MAX-TIME-CUTOFF, see Section II for definitions). For each strategy, we find the optimized cutoff threshold when applied to the nine-node repeater chain protocol from (19) with parameters: $p_{\text{gen}} = 0.1$, $p_{\text{swap}} = 0.4$, $w_0 = 0.98$, $t_{\text{coh}} = 600$. For each cutoff strategy, the plot shows the numerically found waiting time and fidelity distribution for the optimal protocol. We observe that the FIDELITY-CUTOFF strategy yields the largest secret-key rate. However, the DIF-TIME-CUTOFF strategy only performs slightly worse. We observed the same behavior for all other parameter regimes we investigated.

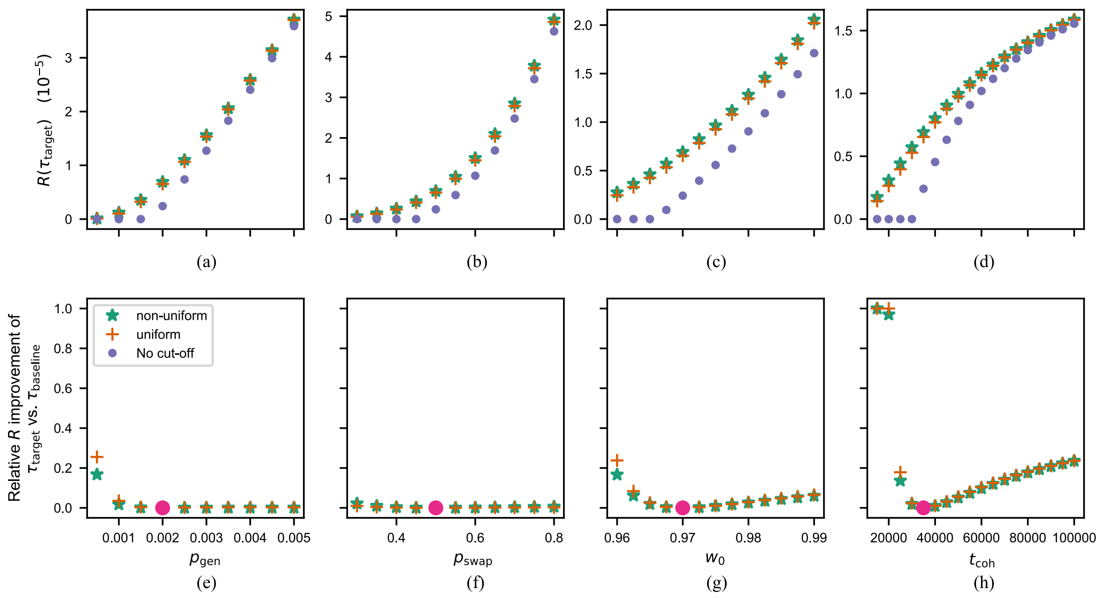

FIGURE 7. Effect of the optimal cutoff (cutoff on the difference in entanglements' production times) on secret-key rate for different hardware parameters, for the nine-node protocol as in (19). We choose a set of parameters as baseline parameters ($p_{\text{gen}} = 0.002$, $p_{\text{swap}} = 0.5$, $w_0 = 0.97$ and $t_{\text{coh}} = 35000$) and in each plot in the figure, we vary only one of the four parameters. The top plots (a)–(d) show the performance of the protocol with optimized cutoffs, where the optimization is implicitly performed for each data point separately. The set of cutoffs we optimize over is either nonuniform (allow for different cutoffs at the three nesting levels of the protocols) or uniform (same cutoff at each level). We observe that the performance difference between uniform and nonuniform cutoffs is small or even negligible. The plots also indicate parameter regimes in which the protocol with the optimal cutoff generates key while its no-cutoff alternative does not (i.e., the no-cutoff has zero secret-key rate). The bottom plots (e)–(h) show relative performance improvement (20) of the optimal cutoff (τ_{target}) for a given data point, versus the optimal cutoff τ_{baseline} for the baseline parameters (see above). The plots show that cutoff performance is most sensitive to coherence time (t_{coh}), while it is least influenced by varying the success probability entanglement swapping (p_{swap}). For a detailed explanation see the main text. Note that the smaller the relative secret-key rate improvement (vertical axis), the closer the performance of τ_{baseline} is to the performance of the optimal τ_{target}, which is why in the plots the best-performing "nonuniform" cutoff shows smaller relative improvement than the best-performing "uniform" cutoff. The purple circles refer to the baseline parameters, for which the relative improvement is 0 by definition.

of parameters that deviates slightly from the baseline values (optimal cutoff τ_{target}), we quantify the sensitivity by their relative performance difference

$$\frac{R(\tau_{\text{target}}) - R(\tau_{\text{baseline}})}{R(\tau_{\text{target}})} \tag{20}$$

where R is the secret-key rate achieved by the repeater protocol. If this relative difference is small, the performance of cutoff is insensitive to the parameter deviation.

In Fig. 7(e)–(h), we plot the relative performance difference for deviations in each of the four hardware parameters separately. We find that the performance of the baseline cutoff is influenced most by variation in coherence time, while it is largely insensitive to change in the swap success

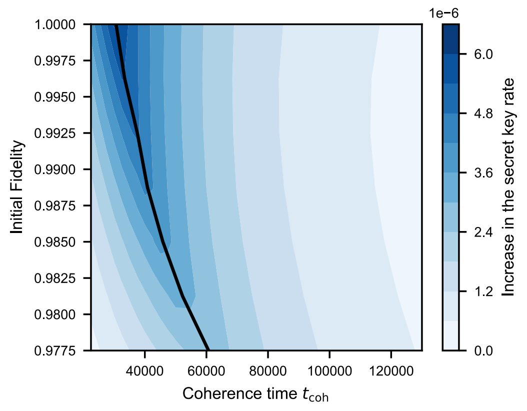

FIGURE 8. **Absolute increase in secret key rate with the optimal cutoff compared to no cutoff as a function of memory coherence time and fidelity of the elementary links (= $(1 + 3w_0)/4$, see Section II), for the 9-node repeater protocols as in (19) where the used cutoff strategy is DIF-TIME-CUTOFF. The black solid line separates the area where the no-cutoff protocol produces no secret key (left of the line) and where its secret-key rate is strictly larger than zero (right of the line). We observe that for the entire parameter range depicted in the figure, cutoffs increase the secret key rate and the absolute improvement is largest for parameters close to the key-producing threshold for the no-cutoff protocol (i.e., close to the black solid line). The plot consists of 126 data points on a grid and the used parameters are $p_{\mathrm{gen}} = 0.001$ and $p_{\mathrm{swap}} = 0.5$. Time unit is the duration of a single elementary link generation attempt.**

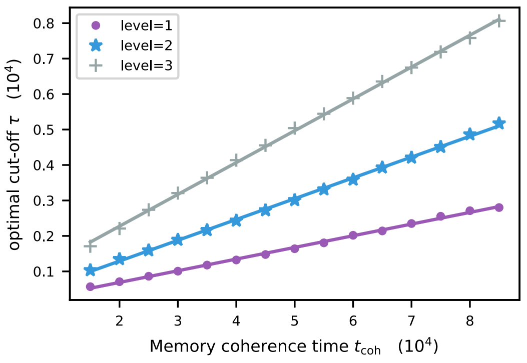

FIGURE 9. **Optimal cutoff as a function of the memory coherence time in the nested 9-node repeater protocols from (19), where the cutoff strategy is (DIF-TIME-CUTOFF). We observe that the numerically found optimal cutoff for different levels is a linear function of the coherence time. Solid lines are linear fits. The hardware parameters used are the same as those for Fig. 7(d). When considering the same protocol on fewer nesting levels (three and five nodes, respectively), we observe similar behavior.**

probability. For the coherence time and the remaining two parameters, the elementary link quality and the success probability of elementary link generation, we distinguish the case where the parameter is improved and the regime where the parameter is made worse. We observe that a worse parameter results in a significant performance difference with the optimal cutoff, while the performance difference is small when the parameter is improved.

We finish by investigating the most influential parameter, the coherence time, in Fig. 9. We observe that the optimal threshold depends approximately linearly on the memory coherence time, which could serve as a heuristic for choosing a performant cutoff.

VI. CONCLUSION

In this article, we optimized the secret key rate over repeater protocols including cutoffs. Our main tool is an algorithm for computing the probability distribution of waiting time and fidelity of the first generated end-to-end link. The algorithm is applicable to a large class of quantum repeater schemes that can include cutoff strategies and distillation. Its runtime is polynomial in the support size of the probability distribution of waiting time.

Our simulations show that the use of the optimal cutoff lowers the hardware quality threshold at which secret key can be generated compared to the no-cutoff alternative. Furthermore, we observed an increase in secret-key rate for the entire regime studied for which the no-cutoff protocol produces nonzero key.

Regarding the choice of cutoff, we find that uniform cutoffs lead to a negligible reduction in the secret key rate compared to the optimal set of cutoffs which differ per nesting level. Moreover, the optimal uniform cutoff is highly sensitive to the quality of the memory, while it is barely influenced by the success probability of swapping. Such sensitivity could guide the heuristic cutoff optimization of more complex protocols.

APPENDIX A
VALIDATION AGAINST A MONTE CARLO ALGORITHM

In this section, we verify that our implementation of the deterministic algorithm presented in Section III is correct by validation against the Monte Carlo sampling algorithm from Brand *et al.* [18]. For all repeater schemes we ran (up to $2^{10} + 1$ nodes for some parameters), we observed good agreement between the waiting time probability distribution and Werner parameter the algorithms computed, which is convincing evidence that our implementation is correct. Fig. 4 depicts the result of a typical run.

What follows is a brief description of the Monte Carlo algorithm from Brand *et al.* [18], including an extension to CUTOFF. Each run of the Monte Carlo algorithm samples a tuple of waiting time and Werner parameter. It is defined recursively by having a dedicated function for each PROTOCOL-UNIT (described below) call the dedicated functions of the two PROTOCOL-UNITS that produce its two input links. The recursion follows the repeater protocol's tree structure (see Fig. 1), resulting in a sampling algorithm of waiting time and Werner parameter of the entire repeater protocol.

The dedicated functions for each of the four PROTOCOL-UNITS are as follows. If the protocol is only a GEN, the Monte Carlo algorithm samples the waiting time from the geometric distribution with parameter p_{gen} and the Werner parameter is the constant w_0. For the other PROTOCOL-UNITS, each of which takes two links as input, the algorithm begins by initializing the total elapsed time $t = 0$. Then, it enters a loop which starts by calling the dedicated functions of the

PROTOCOL-UNITS that produce the two input links, resulting in two samples (t_A, w_A) and (t_B, w_B). The algorithm randomly declares 'success' or 'failure' according to the success probability in Table 1. If it succeeds, the function breaks the loop and outputs $t + \max(t_A, t_B)$ and the resulting Werner parameter $w_{\text{out}}(t_A, w_A, t_B, w_B)$ (see Table 1). If it fails, the total elapsed time t is increased by the waiting time ($\max(t_A, t_B)$ for SWAP and DIST, $\min(t_A, t_B) + \tau$ for CUTOFF) and the function goes back to the start of the loop.

APPENDIX B
ALTERNATIVE ALGORITHM AND ITS COMPLEXITY

In Section III-F, we presented an $\mathcal{O}(t_{\text{trunc}}^2 \log t_{\text{trunc}})$-algorithm for evaluating analytically-derived expressions for the waiting time distribution and average fidelity. Here, we outline how the algorithm can be modified to achieve a complexity reduction to $\mathcal{O}(t_{\text{trunc}} \log t_{\text{trunc}})$ for protocols composed of PROTOCOL-UNITS in Table 1 except for FIDELITY-CUTOFF. Similar to the algorithm from the main text, the modified algorithm consists of two steps: first, evaluating the expressions regarding a single attempt [see (5), (6), and (12)], followed by computing expressions regarding the whole PROTOCOL-UNIT [see (11) and (14)]. We show a complexity reduction for both.

For the first part, we show how to evaluate (5), (6), and (12) in time $\mathcal{O}(t_{\text{trunc}})$, improving on the $\mathcal{O}(t_{\text{trunc}}^2)$ runtime of the algorithm in the main text. Our insight here is that p and $p \cdot w_{\text{out}}$, for SWAP and DIST (see Table 1), can always be written in the form

$$\sum_i f^{(i)}(t_{\text{A}}) \cdot g^{(i)}(t_{\text{B}}) \tag{21}$$

where the $f^{(i)}$ and $g^{(i)}$ are arbitrary functions on the real numbers. For instance, given $t_{\text{A}} \geq t_{\text{B}}$, we can write the success probability of distillation p_{dist} with $f^{(1)}(t_{\text{A}}) = \frac{1}{2}$, $g^{(1)}(t_{\text{B}}) = 1$ and $f^{(2)}(t_{\text{A}}) = \frac{1}{2} w_{\text{A}}(t_A) \exp(-\frac{t_{\text{A}}+t_{\text{c}}}{t_{\text{coh}}})$, $g^{(2)}(t_{\text{B}}) = w_{\text{B}}(t_B) \exp(\frac{t_{\text{B}}-t_{\text{c}}}{t_{\text{coh}}})$. Consequently, each of (5), (6), and (12) can be written in the form

$$\sum_{t_{\text{A}}, t_{\text{B}}: \max(t_{\text{A}}, t_{\text{B}}) = t} \Pr(T_{\text{A}} = t_{\text{A}}, T_{\text{B}} = t_{\text{B}}) \cdot \sum_i f^{(i)}(t_{\text{A}}) g^{(i)}(t_{\text{B}}) \tag{22}$$

which can be rewritten by splitting up the sum in the regime $t_A \geq t_B$ and $t_B > t_A$

$$\sum_{t_{\text{B}}=0}^{t} \Pr(T_{\text{A}} = t, T_{\text{B}} = t_{\text{B}}) \cdot \sum_i f^{(i)}(t) g^{(i)}(t_{\text{B}})$$
$$+ \sum_{t_{\text{A}}=0}^{t-1} \Pr(T_{\text{A}} = t_{\text{A}}, T_{\text{B}} = t) \cdot \sum_i f^{(i)}(t_{\text{A}}) g^{(i)}(t). \tag{23}$$

The first term in (23) can be written as

$$\Pr(T_{\text{A}} = t) \cdot \sum_i f^{(i)}(t) \cdot G^{(i)}(t) \tag{24}$$

where we have defined

$$G^{(i)}(t) = \sum_{t_{\text{B}}=0}^{t} \Pr(T_{\text{B}} = t_{\text{B}}) g^{(i)}(t_{\text{B}}).$$

The expression for the second term in (23) can be found analogously. Computing (24) for all t is now performed by first computing $G^{(i)}(t)$ for all t, which requires linear time in t_{trunc}, and then evaluating (24) for fixed t in constant time. Therefore, the complexity for computing (24) and also for (22) for all t scales as $\mathcal{O}(t_{\text{trunc}})$.

This complexity holds also for protocols with DIF-TIME-CUTOFF and MAX-TIME-CUTOFF, as the cut-off condition appears only as an additional constraint on t_{A} and t_{B} in the sum of (23). For the third cutoff strategy we consider in this work, FIDELITY-CUTOFF, the cutoff condition is not a function of time and therefore the above method does not work.

The second part regards the evaluation of (7) and (13) which is done exactly by the algorithm from the main text in time $\mathcal{O}(t_{\text{trunc}}^2 \log t_{\text{trunc}})$. Here, we give an $\mathcal{O}(t_{\text{trunc}} \log t_{\text{trunc}})$-algorithm which evaluates the equivalent expressions in Fourier space given in Section III-F [see (11) and (14)] with arbitrarily small error. We proceed in two steps. First, we show how to evaluate the expressions in Fourier space exactly in time $\mathcal{O}(t_{\text{trunc}}^2 \log t_{\text{trunc}}^2)$. Then, we show how to achieve a reduction to $\mathcal{O}(t_{\text{trunc}} \log t_{\text{trunc}})$ with an arbitrarily small error.

The expressions in Fourier space [see (11) and (14)] hold for any t in case P_s, P_f and W_s are defined for all $t \geq 0$. However, in the implementation, we truncate the distribution and only have access to them for $0 \leq t < t_{\text{trunc}}$, each stored as an array of length t_{trunc}, and use the discrete Fourier transform defined in (10). The convolution defined in this way is a circular convolution

$$[f_1 \tilde{*} f_2](t) = \sum_{t'=0}^{t} f_1(t - t') \cdot f_2(t')$$
$$+ \sum_{t'=t+1}^{L-1} f_1(L + t - t') \cdot f_2(t') \tag{25}$$

where L is the length of the array and $\tilde{*}$ denotes the circular convolution. The circular convolution introduces discrepancy compared to the linear convolution defined in (8) because $[f_1 \tilde{*} f_2](t) = [f_1 * f_2](t) + [f_1 * f_2](L + t)$. To avoid this, we pad the arrays of P_s, P_f and W_s with zeroes until a length of $L = t_{\text{trunc}}^2$, which is longer than the size of t_{trunc} times convolution of arrays of size t_{trunc} (see equivalent expressions (7) and (13), and the algorithm presented in Section III-F). That is, we set $P_s(t) = 0$ and $P_f(t) = 0$ for $t_{\text{trunc}} \leq t < L = t_{\text{trunc}}^2$. With this setup, the summand in the circular convolution is always 0 for $t' > t$ and it coincides with the linear one. The complexity of the obtained algorithm evaluating (11) and (14) is dominated by one Fourier transform and one inverse Fourier transform on an array of length $\mathcal{O}(t_{\text{trunc}}^2)$. Since a Fourier transform on an array of length L

can be performed in time $\mathcal{O}(L\log L)$, the algorithm has a complexity of $\mathcal{O}(t_{\mathrm{trunc}}^2 \log t_{\mathrm{trunc}}^2)$.

We now show that we can reduce this complexity by zero-padding the arrays only until a length of Ct_{trunc} for some predefined constant C, yielding an exponentially small error

$$\epsilon = \max_t \left(|\Pr(T_{\mathrm{out}} = t) - \Pr(T_{\mathrm{approx}} = t)|\right) \tag{26}$$

in C of the distribution $\Pr(T_{\mathrm{approx}} = t)$ obtained with circular convolution. The resulting algorithm has complexity of $\mathcal{O}(Ct_{\mathrm{trunc}} \log(Ct_{\mathrm{trunc}})) = \mathcal{O}(t_{\mathrm{trunc}} \log t_{\mathrm{trunc}})$.

The motivation behind this reduction is that $\Pr(T_{\mathrm{out}} = t)$ is the sum of all possible sequences of failed attempts [see (7)] and is exponentially decreasing for large t. For a fixed number of attempts k, the probability results from a successful attempt after at least $k-1$ failed attempts. Therefore, it has an occurrence probability of at most $(1-p)^{k-1}$, where p is the success probability for a PROTOCOL-UNIT. To see this mathematically, we use the Young's convolution inequality [35] and obtain

$$\left\| \mathop{*}_{j=1}^{k-1} P_{\mathrm{f}}^{(j)} * P_{\mathrm{s}} \right\| \le \|P_{\mathrm{f}}\|^{k-1} \|P_{\mathrm{s}}\| \le (1-p)^{k-1} \tag{27}$$

where the norm is defined by $\|f(t)\| = \sum_t f(t)$. In addition, note that

$$\left[\mathop{*}_{j=1}^{k-1} P_{\mathrm{f}}^{(j)} * P_{\mathrm{s}}\right](t) = 0 \quad \text{for} \quad t \ge kt_{\mathrm{trunc}}$$

because $P_{\mathrm{f}}(t)$ and $P_{\mathrm{s}}(t)$ are finite arrays of length t_{trunc}. Hence, for $t \ge Kt_{\mathrm{trunc}}$, we only need to consider the terms with $k \ge K+1$, i.e., cases with at least K failed attempts. As a result, we obtain a bound for the probability given in (7) for $t \ge Kt_{\mathrm{trunc}}$

$$\Pr(T_{\mathrm{out}} = t) \le \sum_{k=K+1}^{\infty} (1-p)^{k-1} = \frac{(1-p)^K}{p}.$$

The above expression bounds the distribution with an exponentially decreasing probability with respect to the minimal number of failed attempts, which we now use to bound the error. Because of the circular convolution (25), if we only zero-pad to Ct_{trunc}, the obtained distribution is given by

$$\Pr(T_{\mathrm{approx}} = t) = \sum_{j=0}^{\infty} \Pr(T_{\mathrm{out}} = t + jCt_{\mathrm{trunc}})$$

for $0 \le t < Ct_{\mathrm{trunc}}$. That is, the probability for $t > Ct_{\mathrm{trunc}}$ ($j > 0$) will be added to the first Ct_{trunc} elements, introducing an error in the final result. This error is bounded by

$$\epsilon = \sum_{j=1}^{\infty} \frac{(1-p)^{jC}}{p} \le \frac{(1-p)^C}{p^2} \tag{28}$$

which is exponentially small in C. The same bound can be given in analog for the calculation of $W_{\mathrm{out}}(t)$ defined in (13) by noticing that $W_{\mathrm{s}}(t) \le 1$.

The above bound is only for a single PROTOCOL-UNIT and does not account for the propagation of noise among different levels. However, in practice, as long as one chooses a C large enough so that the error on each array value is below the numerical accuracy, this improved algorithm gives the same result as the algorithm provided in the main text. In addition, the above bound is very loose. In our numerical study, we find that, if the truncation time t_{trunc} is chosen so that more than 99% distribution is covered, it suffices to triple the size of the array during the calculation, i.e., set $C = 3$.

Although in general there exists no efficient algorithm which captures a constant fraction of the probability mass for protocols including a cutoff (see Section III-F), we numerically find that the algorithm outlined above scales polynomially in the number of nodes in some parameter regimes, see Fig. 10.

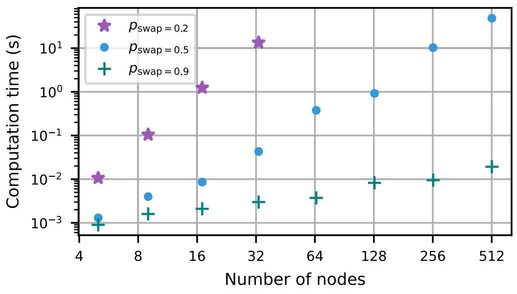

FIGURE 10. Computation time of the algorithm from Appendix B as a function of the number of nodes in the repeater chain using consumer-market hardware (Intel i7-8700 CPU). We plot the computation time for three different p_{swap} and for protocols of the form GEN → (→ CUTOFF → SWAP)^n, similar to (19), where n is the nesting level and the number of nodes is $2^n + 1$. The truncation time is chosen such, that 99% of the probability mass is covered. Note that the plot's axes are both given in logarithmic scale; in such a log-log plot, a polynomial function is represented as a line. The used cutoff strategy is DIF-TIME-CUTOFF and the other parameters used are: $p_{\mathrm{gen}} = 0.1$, $w_0 = 1.0$, $t_{\mathrm{coh}} = 500/p_{\mathrm{swap}}^{n-1}$, $\tau = 42/p_{\mathrm{swap}}^{n-1}$. In this plot, the number of truncation time steps goes up to about 10^6.

APPENDIX C
CALCULATION OF THE SECRET-KEY RATE

Here, we show how we calculate the secret-key rate with truncated waiting time distribution.

One could think of the secret-key rate, computed with finite truncation time $t_{\mathrm{trunc}} < \infty$, as an approximation of the real secret-key rate or, alternatively, as the rate achieved by the following repeater protocol. The protocol starts with the two parties at the end nodes agree on a truncation time t_{trunc}. If up to $t = t_{\mathrm{trunc}}$ the end-to-end link has not been delivered, the protocol terminates and restarts from GEN. Therefore, the number of protocol executions follows the geometric distribution with success probability $p_{\mathrm{tr}} = \Pr(T \le t_{\mathrm{trunc}})$. The waiting time for a failed protocol is t_{trunc} while for a successful one it follows the waiting time distribution $\Pr(T = t)$ for $t < t_{\mathrm{trunc}}$. The average total waiting time is then the sum of the time consumed in failed and successful executions

$$\bar{T} = t_{\mathrm{trunc}} \cdot \left(\sum_{k=1}^{\infty} k \cdot p_{\mathrm{tr}}(1-p_{\mathrm{tr}})^k\right) + \frac{\sum_{t=1}^{t_{\mathrm{trunc}}} t \cdot \Pr(T = t)}{\Pr(T \le t_{\mathrm{trunc}})}. \tag{29}$$

Accordingly, the average Werner parameter is an average over the successful execution

$$\bar{W} = \frac{\sum_{t=1}^{t_{\text{trunc}}} W(t) \cdot \Pr(T = t)}{\Pr(T \leq t_{\text{trunc}})}. \tag{30}$$

With the above equations, we calculate the secret-key rate defined in (17). In this article, we choose heuristically a t_{trunc} such that $\Pr(T \leq t_{\text{trunc}}) \geq 99\%$. With this choice, the difference in the secret key rate between protocols with finite and infinite t_{trunc} is negligibly small.

ACKNOWLEDGMENT

The authors would like to thank S. Brand, K. Goodenough, and F. Rozpędek for helpful discussions.

REFERENCES

[1] H. J. Kimble, "The quantum internet," *Nature*, vol. 453, no. 7198, 2008, Art. no. 1023, doi: 10.1038/nature07127.

[2] S. Wehner, D. Elkouss, and R. Hanson, "Quantum internet: A vision for the road ahead," *Science*, vol. 362, no. 6412, 2018, Art. no. eaam9288, doi: 10.1126/science.aam9288.

[3] H.-J. Briegel, W. Dür, J. I. Cirac, and P. Zoller, "Quantum repeaters: The role of imperfect local operations in quantum communication," *Phys. Rev. Lett.*, vol. 81, pp. 5932–5935, Dec. 1998, doi: 10.1103/PhysRevLett.81.5932.

[4] W. J. Munro, K. Azuma, K. Tamaki, and K. Nemoto, "Inside quantum repeaters," *IEEE J. Sel. Topics Quantum Electron.*, vol. 21, no. 3, pp. 78–90, May 2015, doi: 10.1109/jstqe.2015.2392076.

[5] S. Muralidharan, L. Li, J. Kim, N. Lütkenhaus, M. D. Lukin, and L. Jiang, "Optimal architectures for long distance quantum communication," *Sci. Rep.*, vol. 6, 2016, Art. no. 20463, doi: 10.1038/srep20463.

[6] O. A. Collins, S. D. Jenkins, A. Kuzmich, and T. A. B. Kennedy, "Multiplexed memory-insensitive quantum repeaters," *Phys. Rev. Lett.*, vol. 98, Feb. 2007, Art. no. 060502, doi: 10.1103/PhysRevLett.98.060502.

[7] L. Praxmeyer, "Reposition time in probabilistic imperfect memories," 2013. [Online]. Available: https://arxiv.org/abs/1309.3407

[8] N. Kalb *et al.*, "Entanglement distillation between solid-state quantum network nodes," *Science*, vol. 356, no. 6341, pp. 928–932, Jun. 2017, doi: 10.1126/science.aan0070.

[9] F. Rozpędek *et al.*, "Near-term quantum-repeater experiments with nitrogen-vacancy centers: Overcoming the limitations of direct transmission," *Phys. Rev. A*, vol. 99, May 2019, Art. no. 052330, doi: 10.1103/PhysRevA.99.052330.

[10] F. Rozpędek *et al.*, "Parameter regimes for a single sequential quantum repeater," *Quantum Sci. Technol.*, 2018, doi: 10.1088/2058-9565/aab31b.

[11] S. Santra, L. Jiang, and V. S. Malinovsky, "Quantum repeater architecture with hierarchically optimized memory buffer times," *Quantum Sci. Technol.*, vol. 4, no. 2, Mar. 2019, Art. no. 025010, doi: 10.1088/2058-9565/ab0bc2.

[12] K. Chakraborty, F. Rozpędek, A. Dahlberg, and S. Wehner, "Distributed routing in a quantum internet," 2019. [Online]. Available: http://arxiv.org/abs/1907.11630

[13] P. van Loock *et al.*, "Extending quantum links: Modules for fiber-and memory-based quantum repeaters," *Adv. Quantum Technol.*, vol. 3, no. 11, 2020, Art. no. 1900141, doi: 10.1002/qute.201900141.

[14] F. Schmidt and P. van Loock, "Memory-assisted long-distance phase-matching quantum key distribution," *Phys. Rev. A*, vol. 102, no. 4, 2020, Art. no. 042614, doi: 10.1103/PhysRevA.102.042614.

[15] S. Khatri, C. T. Matyas, A. U. Siddiqui, and J. P. Dowling, "Practical figures of merit and thresholds for entanglement distribution in quantum networks," *Phys. Rev. Res.*, vol. 1, Sep. 2019, Art. no. 023032, doi: 10.1103/PhysRevResearch.1.023032.

[16] E. Shchukin, F. Schmidt, and P. van Loock, "Waiting time in quantum repeaters with probabilistic entanglement swapping," *Phys. Rev. A*, vol. 100, Sep. 2019, Art. no. 032322, doi: 10.1103/PhysRevA.100.032322.

[17] Y. Wu, J. Liu, and C. Simon, "Near-term performance of quantum repeaters with imperfect ensemble-based quantum memories," *Phys. Rev. A*, vol. 101, Apr. 2020, Art. no. 042301, doi: 10.1103/PhysRevA.101.042301.

[18] S. Brand, T. Coopmans, and D. Elkouss, "Efficient computation of the waiting time and fidelity in quantum repeater chains," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 3, pp. 619–639, Mar. 2020, doi: 0.1109/JSAC.2020.2969037.

[19] M. Żukowski, A. Zeilinger, M. A. Horne, and A. K. Ekert, ""Event-ready-detectors" bell experiment via entanglement swapping," *Phys. Rev. Lett.*, vol. 71, pp. 4287–4290, Dec. 1993, doi: 10.1103/PhysRevLett.71.4287.

[20] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters, "Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels," *Phys. Rev. Lett.*, vol. 70, pp. 1895–1899, Mar. 1993, doi: 10.1103/PhysRevLett.70.1895.

[21] C. H. Bennett, G. Brassard, S. Popescu, B. Schumacher, J. A. Smolin, and W. K. Wootters, "Purification of noisy entanglement and faithful teleportation via noisy channels," *Phys. Rev. Lett.*, vol. 76, pp. 722–725, Jan. 1996, doi: 10.1103/PhysRevLett.76.722.

[22] D. Deutsch, A. Ekert, R. Jozsa, C. Macchiavello, S. Popescu, and A. Sanpera, "Quantum privacy amplification and the security of quantum cryptography over noisy channels," *Phys. Rev. Lett.*, vol. 77, pp. 2818–2821, Sep. 1996, doi: 10.1103/PhysRevLett.77.2818.

[23] W. Dür, H.-J. Briegel, J. I. Cirac, and P. Zoller, "Quantum repeaters based on entanglement purification," *Phys. Rev. A*, vol. 59, pp. 169–181, Jan. 1999, doi: 10.1103/PhysRevA.59.169.

[24] R. F. Werner, "Quantum states with Einstein-Podolsky-Rosen correlations admitting a hidden-variable model," *Phys. Rev. A*, vol. 40, pp. 4277–4281, Oct. 1989, doi: 10.1103/PhysRevA.40.4277.

[25] W. Dür and H. Briegel, "Entanglement purification and quantum error correction," *Rep. Prog. Phys.*, vol. 70, no. 8, 2007, Art. no. 1381, doi: 10.1088/0034-4885/70/8/R03.

[26] V. V. Kuzmin, D. Vasilyev, N. Sangouard, W. Dür, and C. Muschik, "Scalable repeater architectures for multi-party states," *NPJ Quantum Inf.*, vol. 5, no. 1, pp. 1–6, 2019, doi: 10.1038/s41534-019-0230-3.

[27] V. V. Kuzmin and D. V. Vasilyev, "Diagrammatic technique for simulation of large-scale quantum repeater networks with dissipating quantum memories," *Phys. Rev. A*, vol. 103, no. 3, 2021, Art. no. 032618, doi: 10.1103/PhysRevA.103.032618.

[28] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Math. Comput.*, vol. 19, no. 90, pp. 297–301, Apr. 1965, doi: 10.2307/2003354.

[29] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," in *Proc. IEEE Int. Conf. Comput., Syst. Signal Process.*, Bangalore, India, vol. 175, 1984, pp. 175–179, doi: 10.1016/j.tcs.2014.05.025.

[30] P. W. Shor and J. Preskill, "Simple proof of security of the BB84 quantum key distribution protocol," *Phys. Rev. Lett.*, vol. 85, pp. 441–444, Jul. 2000, doi: 10.1103/PhysRevLett.85.441.

[31] H.-K. Lo, H. F. Chau, and M. Ardehali, "Efficient quantum key distribution scheme and a proof of its unconditional security," *J. Cryptol.*, vol. 18, no. 2, pp. 133–165, 2005, doi: 10.1007/s00145-004-0142-y.

[32] R. Storn and K. Price, "Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces," *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, 1997, doi: 10.1023/A:1008202821328.

[33] P. Virtanen *et al.*, "SciPy 1.0: Fundamental algorithms for scientific computing in python," *Nat. Methods*, vol. 17, pp. 261–272, 2020, doi: 10.1038/s41592-019-0686-2.

[34] "Optimization of cut-offs for repeater chains," 2020. [Online]. Available: https://github.com/BoxiLi/repeater-cut-off-optimization

[35] V. I. Bogachev, *Measure Theory*. Berlin, Germany: Springer, 2007. [Online]. Available: https://www.springer.com/gp/book/9783540345138