Spanning-tree-packing protocol for conference key propagation in quantum networks

Anton Trushechkin,[1, 2, *] Hermann Kampermann,[1] and Dagmar Bruß[1]
[1] *Heinrich Heine University Düsseldorf, Faculty of Mathematics and Natural Sciences, Institute for Theoretical Physics III, Universitätsstr. 1, Düsseldorf 40225, Germany*
[2] *Steklov Mathematical Institute of Russian Academy of Sciences, Gubkina 8, Moscow 119991, Russia*

(Dated: June 12, 2025)

We consider a network of users connected by pairwise quantum key distribution (QKD) links. Using these pairwise secret keys and public classical communication, the users want to generate a common (conference) secret key at the maximal rate. We propose an algorithm based on spanning tree packing (a known problem in graph theory) and prove its optimality. This algorithm enables optimal conference key generation in modern quantum networks of arbitrary topology. Additionally, we discuss how it can guide the optimal placement of new bipartite links in the network design.

I. INTRODUCTION

Quantum key distribution (QKD) is able to provide secure communication [1–4]. This is important in view of the progress in quantum computation observed now [5, 6], but also in view of possible inventions in the field of efficient algorithms that can threaten conventional cryptography. Originally, QKD protocols allow two users to establish a common secret key (bitstring). However, the real world is often interested not only in point-to-point communication, but in the communication of many users – network communication. For the widespread adoption of quantum communication technologies, a well-developed theory of quantum networks is essential.

Several theoretical and experimental works have investigated networks of nodes connected by sources of bipartite or multipartite entangled states [7–17]. Multipartite QKD, or quantum conference key agreement, which uses multipartite entangled states has been suggested and analyzed in Refs. [18–25]. In the future, such networks are hoped to constitute a "quantum internet" [26–33].

However, in the near future one would expect technologically simpler networks of nodes connected by pairwise QKD, not requiring the creation and manipulation of multipartite entangled states. Such bipartite QKD networks already exist in a number of countries [34]. Protocols for classical information transmission in QKD networks are under active development [35–37] and the general structure of future QKD networks is being investigated [38, 39].

Examples of graphs representing such networks are given in Fig. 1. Vertices correspond to users and an edge between two nodes means that the corresponding pair of users is connected by a point-to-point (bipartite) QKD link. We assume that all QKD links can operate in parallel, although alternative scenarios may also be considered.

A variety of tasks can be studied for such networks. In this work, we focus on the task of generating a common

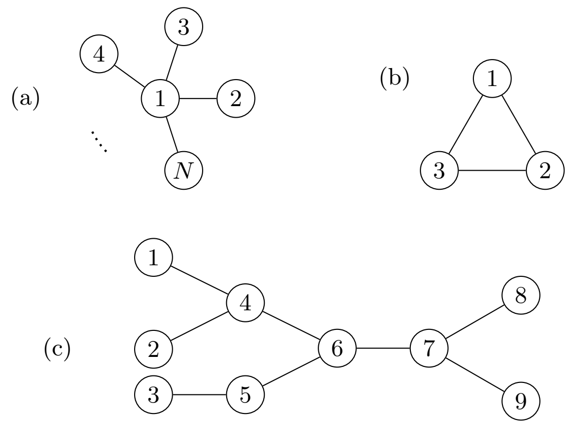

Figure 1. Examples of QKD networks: a star network with N users (a), the triangle network (b) and a tree network (c). Vertices (nodes) of a graph represent users and edges represent (bipartite) QKD links between them.

(conference) secret key. It is well-known (see, e.g., [19]) that users having a connected network of bipartite keys can agree on a common (conference) secret key using classical communication (XOR operations). The question we address is to do it optimally, i.e., to generate the conference key at the maximal rate given the bipartite rates in the network. We refer to this task as optimal *conference key propagation* because we want to propagate pairwise secret keys over the network, resulting in a conference key.

We propose a protocol of conference key propagation for networks of arbitrary topologies based on the spanning tree packing (STP). Optimal spanning tree packing is a known and efficiently solvable problem in graph theory [40–42]. We prove that the optimal spanning tree packing gives the optimal conference key propagation in this setting and can be used in existing QKD networks.

As further applications of this result, we discuss the derivation of simple upper bounds for the conference key rate for networks with bottleneck structures. We also explore how this framework can guide the optimal placement of new QKD links in network design.

Note that other approaches to quantum conference

[*] anton.trushechkin@hhu.de

key agreement without direct use of multipartite entangled states include the MDI (measurement-device-independent) approach, where the multipartite entangled state is postselected a posteriori in a system of detectors [43–52], and simultaneous delivering of the copies of the same quantum state from a sender to many receivers [53].

Our paper is organized as follows. In Sec. II, we state the problem. In Sec. III, we describe the protocol starting from simple examples. In Sec. IV, we formulate a theorem of its optimality. In Sec. V, we discuss the upper bounds on the conference key rate originating from complex bottleneck structures and a related problem of network optimization. In Sec. VI, we give the formula for the universally composable security parameter $\varepsilon_{\rm conf}$ for the conference key generated by the proposed STP protocol. Since the existing efficient algorithm solving the spanning tree packing contains many steps [42], in Appendix A we propose a simpler heuristic algorithm for this problem used in the examples of this paper. In Appendix B, we give a proof of the main theorem.

II. PROBLEM STATEMENT

A network is represented as a graph $(\mathcal{V},\mathcal{E})$, where $\mathcal{V}=\{1,\dots,N\}$ is a set of vertices (nodes) associated with users and $\mathcal{E}$ is the set of edges associated with bipartite QKD links between the users. We always assume that the graph is connected, i.e., there is a path between any pair of vertices.

We assume that all QKD links $e\in\mathcal{E}$ operate in parallel for time T and generate the keys K_e of lengths L_e. Formally, K_e are random variables taking values on $\{0,1\}^{L_e}$. Each launch of all QKD links for time T will be called "a round". Then, the bipartite secret key rates are $r_e=L_e/T$. If after n rounds (which take the total time nT), we generate a conference key of the length $L_{\rm conf}$, then the conference key rate is given by $r_{\rm conf}=L_{\rm conf}/(nT)$.

In practice, the bipartite secret key rates are typically integers (e.g., certain number of bits per second), which will be assumed for simplicity. However, some of our formulas and calculations do not rely on the property of r_e being integer and are valid also for rational or real r_e.

The task is stated as follows: given a graph $(\mathcal{V},\mathcal{E})$ and bipartite rates $\{r_e\}_{e\in\mathcal{E}}$, build a conference key propagation protocol with the maximally possible rate $r_{\rm conf}$. For a formal definition, see Eq. (10) in Sec. IV.

Thus, we deal with a weighted graph with the edge weights given by the bipartite key rates r_e, see Fig. 2(a). Since the weights r_e are integers, we can treat the graph as a multigraph with the multiplicity of edges r_e, see Fig. 2(b).

It is known that keys generated by QKD deviate from being perfectly secure and this deviation is characterised by a security parameter ε [54, 55]. So, in general, each edge e is characterized by a security parameter ε_e. We start with perfectly secure bipartite keys and return to the practical case of non-zero ε_e in Sec. VI.

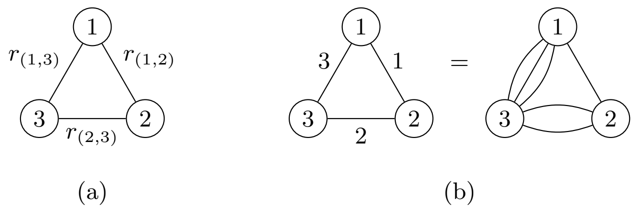

Figure 2. Quantum QKD network as a weighted graph with bipartite secret key rates r_e as the weights (a). In the case of integer weights, it can be alternatively represented as the multigraph where edge multiplicities correspond to the weights (b).

III. DESCRIPTION OF THE PROTOCOL

A. Simple case: Tree networks

Before we formulate the general protocol, let us start with simple examples. Consider the star network depicted in Fig. 1(a), where the central node 1 is connected to $N-1$ other nodes $2,\dots,N$ and there are no more connections in the network. Thus, $\mathcal{E}=\{(1,2),(1,3),\dots,(1,N)\}$. In such networks, user 1 is distinguished and often called "Alice", while all others are "Bobs".

Also, for simplicity, let us assume that all bipartite secret key rates are $r_{1j}=1$, $j=2,\dots,N$, i.e., each QKD link generates one bipartite secret bit per round. The conference key is generated as follows, see, e.g., Ref. [19]. The participants publicly choose which of the bipartite keys will be the conference key. For example, they choose the key K_{12} as the future conference key. Then Alice encrypts K_{12} as the message to the Bobs $3,\dots,N$ by the one-time pad encryptions using the corresponding bipartite keys $K_{(1,3)},K_{(1,4)},\dots,K_{(1,N)}$. Namely, Alice publicly announces the bitstrings

$$\begin{aligned}&K_{(1,2)}\oplus K_{(1,3)},\\&K_{(1,2)}\oplus K_{(1,4)},\\&\dots\\&K_{(1,2)}\oplus K_{(1,N)}.\end{aligned}\tag{1}$$

Here $\oplus$ is XOR (addition modulo 2). The users $3,4,\dots,N$ can decrypt the ciphertext (apply the operation $\oplus K_{(1,j)}$ for the corresponding j again) to obtain the bit $K_{(1,2)}$, which then can be used as a conference key. Since we have generated one bit of the conference key with one round of bipartite QKD protocols, $r_{\rm conf}=1$. This is a known obvious solution.

If r_e are different integers, then the conference key rate is upper bounded by the minimal bipartite rate $\min r_e$. Keeping $\min r_e$ bits in each of the keys and repeating the described procedure gives $r_{\rm conf}=\min r_e$.

This protocol can be easily generalized to any connected graph without cycles, i.e., a tree. Let us again assume that all bipartite secret key rates are $r_e = 1$. The protocol is illustrated in Fig. 3 for the graph given in Fig. 1(c). Again, the participants publicly agree on which of the bipartite keys $K_{\bar{e}}$ will be the conference key. Since there are no cycles in the graph, removal of the edge $\bar{e} = (i, j)$ from the graph breaks the tree graph into two non-connected subtrees. In Fig. 3, the removal of the edge $(6, 7)$ breaks the graph into two subtrees depicted by blue and green with vertices 6 and 7 as the roots. Setting i and j as the roots of these two subtrees, we can make the graph oriented and thus to construct unique paths from i or j to every vertex of the corresponding subtree.

In our example, vertex 6 encrypts the message $K_{\bar{e}}$ for its children in the corresponding subtree (vertices 4 and 5) using the one-time pad and the corresponding bipartite secret keys. Then the vertices that are not leafs in the subtrees do the same for their children in the subtrees. In our example, vertices 4 and 5 encrypt the future conference key $K_{(5,6)}$ for the users 1, 2, and 3 using the bipartite keys $K_{(1,4)}$, $K_{(2,4)}$, and $K_{(3,5)}$. Vertex 7 and its descendants act in an analogous way. Thus, the conference key $K_{(5,6)}$ propagates along the whole network.

In this version of the protocol, vertices 4 and 5 have to wait for messages from vertex 6 in order to send their messages for vertices 1, 2, and 3. Let us give an equivalent protocol, which is single-round, i.e., each participant makes public announcements independently on other announcements. After all announcements have been made, each participant can recover the conference key.

Namely, each vertex α except the roots i and j of the subtrees (6 and 7 in our example) has exactly one incoming (directed) edge in_α and a (possibly empty) set of outcoming edges out_α. Also, by definition, we put $\mathrm{in}_i = j$ and $\mathrm{in}_j = i$. Then each user k with non-empty out_k publicly announces the bitstrings

$$C_e^{(\alpha)} = K_{\mathrm{in}_\alpha} \oplus K_e \tag{2}$$

for all $e \in \mathrm{out}_\alpha$. That is, each user encrypts the bitstring K_{in_α} using one-time pad and keys K_e, $e \in \mathrm{out}_\alpha$, and announces the ciphertexts. Then, each user, step by step, going back from its vertex to i and j, decrypt the ciphertexts and finally recovers the bitstring $K_{\bar{e}} = K_{(i,j)}$, which can be used as a conference key.

In our example, the announcements are:

$$\begin{aligned} C_{(4,6)}^{(6)} &= K_{(6,7)} \oplus K_{(4,6)}, & C_{(5,6)}^{(6)} &= K_{(6,7)} \oplus K_{(5,6)}, \\ C_{(1,4)}^{(4)} &= K_{(4,6)} \oplus K_{(1,4)}, & C_{(2,4)}^{(4)} &= K_{(4,6)} \oplus K_{(2,4)}, \\ & & C_{(3,5)}^{(5)} = K_{(5,6)} \oplus K_{(3,5)}, & \\ C_{(7,8)}^{(7)} &= K_{(6,7)} \oplus K_{(7,8)}, & C_{(7,9)}^{(7)} &= K_{(6,7)} \oplus K_{(7,9)}. \end{aligned}$$

Then, for example, user 1 recovers the conference key rate calculating

$$K_{(1,4)} \oplus C_{(1,4)}^{(4)} \oplus C_{(4,6)}^{(6)} = K_{(6,7)}.$$

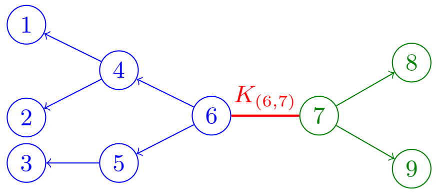

Figure 3. Conference key propagation for the tree graph from Fig. 1(c). The users agree whose bipartite key will be the conference key. Here: $K_{(6,7)}$. Removal of the corresponding edge breaks the graph into two non-connected subgraphs (subtrees, depicted by the blue and green colors) with vertices 6 and 7 as the roots. The propagation of the key $K_{(6,7)}$ proceeds according to the arrows.

As in the case of the star graph, $L_{\mathrm{conf}} = 1$ and $r_{\mathrm{conf}} = 1$. If the bipartite keys are perfectly secure, then, due to perfect security of the one-time pad and the chain-like structure of encryptions (2), the conference key is also perfectly secure. For arbitrary bipartite keys r_e, we can apply the same reasonings as for the star graph and conclude that $r_{\mathrm{conf}} = \min r_e$.

Thus, we arrive at the following observation:

Observation 1. *A tree of bipartite secret bits leads to one conference secret bit.*

For optimal conference key propagation, a general problem is to pack as many trees into a given graph as possible, which is a well-known problem in graph theory. Before we give precise formulations of this problem in Sec. III C, we consider one more instructive simple example in the next subsection.

B. Triangle network

Now consider the simplest network with a cycle: a triangle network depicted in Fig 1(b). For simplicity, we still assume that all bipartite key rates are equal to one. How can the users agree on a conference secret key? Of course, they can ignore, for example, the QKD link $(2, 3)$ and reduce the problem to the star graph case described in Sec. III A, which leads to the conference key rate $r_{\mathrm{conf}} = 1$. In general, every graph with cycles can be reduced to a graph without cycles by removing a subset of edges. However, ignoring QKD links might be non-optimal.

Let us propose an improved protocol. Let us launch two rounds of bipartite QKD links. Then, each link generates two secret bits $K_{e,1}$ and $K_{e,2}$. Then the participants 1,2 and 3 announce, respectively,

$$\begin{aligned} C^{(1)} &= K_{(1,2),1} \oplus K_{(1,3),1}, \\ C^{(2)} &= K_{(1,2),2} \oplus K_{(2,3),1}, \\ C^{(3)} &= K_{(1,3),2} \oplus K_{(2,3),2}, \end{aligned} \tag{3}$$

respectively. This information is sufficient for all participants to recover all bipartite keys. Then, they can decide to accept, e.g., $K_{(1,2),1}$, $K_{(1,2),2}$, and $K_{(1,3),2}$ as three bits for the conference key. That is, we have generated $L_{\rm conf}=3$ bits of the conference key using $n=2$ rounds of bipartite links, hence, the conference key rate is

$$r_{\rm conf}=\frac{L_{\rm conf}}{n}=\frac{3}{2}, \tag{4}$$

which is greater than the value $r_{\rm conf}=1$ obtained by ignoring one of the links. Since Eve does not know $K_{(1,3),1}$, $K_{(2,3),1}$, and $K_{(2,3),2}$, she has no information about the conference key. Thus, we have obtained a better protocol for conference key propagation, which uses all the bipartite QKD links. In the following, we generalize and formalize the scheme given in this example.

C. General formulation of the STP-based conference key propagation protocol

The essence of the protocol for the last example is depicted in Fig 4 (top row). For the graph $(\mathcal{V},\mathcal{E})$ with the weights (bipartite key rates) r_e, consider the sequence of mutligraphs (i.e., graphs where multiple edges between the same pair of vertices are allowed) $(\mathcal{V},\mathcal{E},\{nr_e\}_{e\in\mathcal{E}})$, $n=1,2,\dots$, where nr_e (numbers of bipartite bits generated in n rounds) are the edge multiplicities, see Fig. 2(b). Recall that a subgraph of a graph or a multigraph is called a spanning tree if it is a tree (i.e., connected graph without cycles) and contains all vertices of the original graph. We can reformulate Observation 1 as follows:

Observation 2. *A spanning tree of bipartite secret bits in the multigraph $(\mathcal{V},\mathcal{E},\{nr_e\})$ leads to one conference secret bit.*

If we wish to maximize the conference key rate, we need to pack as many spanning trees into the multigraph $(\mathcal{V},\mathcal{E},\{nr_e\})$ in the following sense:

$$r_{\rm conf}=\max_{k,n}\left\{\frac{k}{n}\;\middle|\;\begin{matrix}(\mathcal{V},\mathcal{E},\{nr_e\})\text{ contains }k\text{ edge-disjoint}\\ \text{spanning trees}\end{matrix}\right\}. \tag{5}$$

That is, if we generated the conference key of length k using n rounds of parallel bipartite QKD links, then the conference key rate is k/n and we want to maximize this quantity. In our example with the triangle network depicted in Fig 4 (top row), the multigraph $(\mathcal{V},\mathcal{E},\{2r_e\})$ can be partitioned into three edge-disjoint spanning trees, hence the conference key rate is $3/2$. Fig. 4 shows the spanning tree packing for other graphs also under the assumption of all the bipartite secret key rates are equal to one.

The problem of finding the maximal number of edge-disjoint spanning trees in a graph or multigraph is called the *spanning tree packing problem* and is well-known in graph theory [40–42]. Its solution, i.e., the maximum itself is called the *spanning tree packing number.* More precisely, the standard formulation of the spanning tree packing assumes that the multigraph is fixed, which corresponds to the case when the maximization in Eq. (5) is performed only over k for a fixed n. Since, in our case, we are free to choose the number of bipartite rounds for the generation of the conference key, we use a slightly different formulation.

The security of the conference key obtained by spanning tree packing relies on the security of the one-time pad and the requirement that spanning trees must be edge-disjoint, hence, each bipartite key is used only once.

Importantly, an efficient algorithm of finding optimal spanning tree packing exists [42]. Also, in Appendix A, we give a simple heuristic algorithm. Formally, in contrast to the algorithm from Ref. [42], it has at least exponential time complexity. Nevertheless, in practice, it allows one to easily find optimal spanning tree packings for small graphs or graphs with recognizable structures (certain patterns of edges, recognizable clusters, etc.) and was used for all examples in this paper.

The spanning-tree-packing number is given by the Nash-Williams-Tutte theorem [40–42]. Consider a vertex partition $P=\{\mathcal{V}_1,\dots,\mathcal{V}_p\}$, so that $\mathcal{V}=\mathcal{V}_1\sqcup\ldots\sqcup\mathcal{V}_p$ (recall that the symbol $\sqcup$ denotes the union of disjoint sets), $p\geq 2$, and all subsets $\mathcal{V}_i\subset\mathcal{V}$ are non-empty. Denote $\mathcal{E}(P)$ the set of cross-edges in the graph $(\mathcal{V},\mathcal{E})$, i.e., the edges whose vertices belong to different subsets in the partition P. Also, according to standard mathematical notations, $|P|=p$ is the number of elements (vertex subsets) in the partition P. Then, the multigraph $(\mathcal{V},\mathcal{E},\{nr_e\})$ has exactly

$$k\equiv L^{(n)}_{\rm conf}=\min_{\substack{\text{vertex}\\ \text{partitions }P}}\left\lfloor\frac{1}{|P|-1}\sum_{e\in\mathcal{E}(P)}nr_e\right\rfloor \tag{6}$$

edge-disjoint spanning trees, which corresponds to the conference key length $L^{(n)}_{\rm conf}$ and the conference key rate $r^{(n)}_{\rm conf}=L^{(n)}_{\rm conf}/n$. For large enough n, the argument of the floor function $\lfloor\cdot\rfloor$ is integer and, thus, this function can be removed. This leads to the conference key rate

$$r_{\rm conf}=\min_{\substack{\text{vertex}\\ \text{partitions }P}}\frac{1}{|P|-1}\sum_{e\in\mathcal{E}(P)}r_e. \tag{7}$$

This is indeed the solution of the optimization problem (5): As we argued, the rate given by the right-hand side of Eq. (7) is achieved by choosing a proper n and the rate cannot be greater than the right-hand side of Eq. (7) because $r^{(n)}_{\rm conf}\leq r_{\rm conf}$ for all n. In this paper, we will refer to Eq. (7) as the Nash-Williams–Tutte formula.

IV. OPTIMALITY OF THE STP-BASED PROTOCOL

We have considered the problem of the optimal spanning tree packing. Can we propose another conference

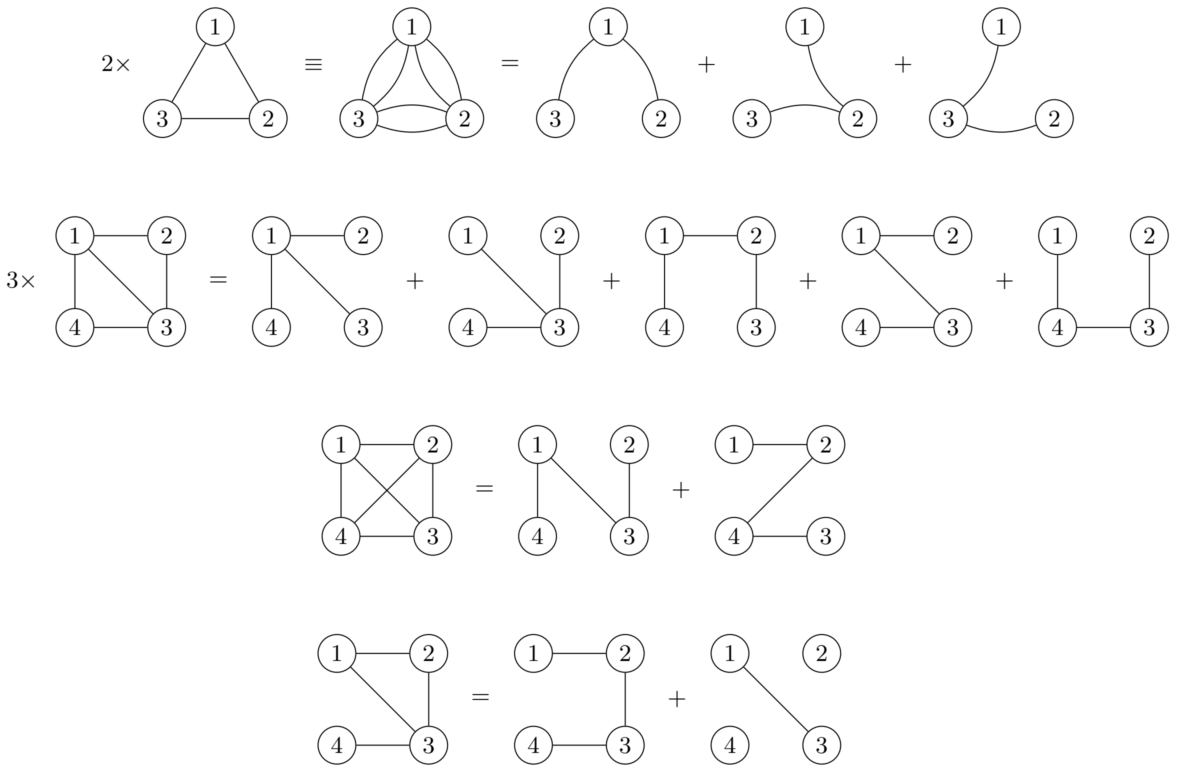

Figure 4. Spanning trees packings in different graphs with equal edge weights (bipartite QKD rates) for all edges. Here it is convenient to think that an edge in a graph represents a bipartite key with a fixed length L. Then multiple QKD sessions (factors in front of of the graphs) give multiple edges between the same vertices. One spanning tree correspond to a conference key of the length L. Thus, the problem is to find an optimal *spanning tree packing*, i.e. a set of edge-disjoint spanning trees such that the ratio of the number of edge-disjoint spanning trees to the number of QKD sessions (per each bipartite link) is maximal. The last term in the last line is not a spanning tree, but a leftover edge. The operations $\times$ and $+$ are understood here in the sense of edge multiplicities in a multigraph. If all bipartite secret key rates are equal to one ($L = 1$), then the conference key rates r_{conf} for these examples (from top to bottom) are 3/2, 5/3, 2, and 1.

key propagation protocol (not necessarily related to the spanning-tree packing) for a network of bipartite QKD links that would give better results than Eqs. (5) and (7)? The answer is negative: optimal spanning-tree-packing is also optimal among all possible conference key propagation protocols. In order to prove it rigorously, we need to give a formal problem statement.

The conference key propagation problem is a particular case of secret key distillation in classical networks [56]. Let, again, K_e, $e \in \mathcal{E}$, be secret keys, i.e., random variables uniformly distributed on the sets $\{0,1\}^{r_e}$. Denote by $\mathcal{E}_i$ the set of edges incident to the vertex $i \in \{1,\dots,N\}$ and by

$$X_i = (K_e)_{e\in\mathcal{E}_i} \tag{8}$$

the random variable obtained by the user i, i.e., the collection of bipartite keys of all its connections. The participant starts from n independent and identically distributed (iid) repetitions of K_e and, thus, X_i. Denote n iid repetitions of X_i as X_i^n. Each participant i generates also local randomness, i.e., a random variable R_i from an arbitrary (finite) alphabet. Then the participants publicly announce (classical) messages $C_1,\dots,C_M$, where each C_j is a function of (X_i^n, R_i) for $i = (j-1 \bmod N)+1$ and of the previous messages $C_1,\dots,C_{j-1}$. Denote $C = (C_1,\dots,C_M)$. Finally, each participant i calculates its version of the conference key $K_{\mathrm{conf}}^{(i)} \in \{0,1\}^m$ as a function of (X_i^N, R_i, C). Actually, this is a protocol with classical local operations and public communication (CLOPC) with a finite number of communication rounds. Following the notations of Ref. [57], denote it as $\mathrm{CLOPC}_{\mathbb{N}}$, where the subscript means that the number of rounds can be arbitrary large, but finite.

Denote by $P_{X_1\dots X_N}$ and $P_{K_{\mathrm{conf}}^{(1)}\dots K_{\mathrm{conf}}^{(N)}C}$ the original distribution of $X_1,\dots,X_N$ and the final distribution of the version of the conference key and the communication, respectively. Then

$$\Lambda(P_{X_1\dots X_N}^{\otimes n}) = P_{K_{\mathrm{conf}}^{(1)}\dots K_{\mathrm{conf}}^{(N)}C} \tag{9}$$

for some $\Lambda \in \mathrm{CLOPC}_{\mathbb{N}}$. Denote also by P_C the marginal distribution of C. The *conference key propagation capacity* is defined as

$$\overline{r}_{\mathrm{conf}} = \lim_{\varepsilon\to 0} \sup_{\substack{n,m,\\ \Lambda\in\mathrm{CLOPC}_{\mathbb{N}}}} \Big\{\frac{m}{n}\Big| \frac{1}{2}\|\Lambda(P^{\otimes n}_{X_1\dots X_N}) - P^{(m),\,\mathrm{ideal}}_{K^{(1)}_{\mathrm{conf}}\dots K^{(N)}_{\mathrm{conf}}} \otimes P_C\|_1 \le \varepsilon\Big\}, \quad (10)$$

where

$$P^{(m),\,\mathrm{ideal}}_{K^{(1)}_{\mathrm{conf}}\dots K^{(N)}_{\mathrm{conf}}}(k_1,\dots,k_N) = \begin{cases} 2^{-m}, & k_1 = \dots = k_N, \\ 0, & \text{otherwise} \end{cases} \quad (11)$$

and $\frac{1}{2}\|\cdot\|_1$ denotes the total variational distance of two probability distributions. Definition (10) is given for a general multipartite probability distribution $P_{X_1\dots X_N}$, but the special form (8) of X_i corresponds to the bipartite network structure.

Now we can formulate our main theorem.

Theorem 1. *The conference key propagation capacity (10) for an arbitrary bipartite network structure (8) is equal to the spanning-tree-packing number (7).*

The proof is given in Appendix B. It is based on combining the theorem by Csiszár and Narayan [56] about the conference secret key capacity (for classical networks) with the Nash-Williams–Tutte formula (7). The main technical ingredient is the following proposition:

Proposition 1. *Let a graph $(\mathcal{V},\mathcal{E})$ with N vertices and non-negative real edge weights $\{r_e\}_{e\in\mathcal{E}}$ be given. The number r_{conf} given by Eq. (7) is equal to the number Z obtained from the following linear program:*

$$Z = \sum_{e\in\mathcal{E}} r_e - R_{\mathrm{CO}}, \quad (12a)$$

$$R_{\mathrm{CO}} = \min_{R_1,\dots,R_N} \sum_{i=1}^{N} R_i \quad \textit{such that} \quad (12b)$$

$$\sum_{i\in I} R_i \ge \sum_{e\in\mathcal{E}(I)} r_e, \quad \forall I \subsetneq [N]. \quad (12c)$$

Here $[N] = \{1,\dots,N\}$ and $\mathcal{E}(I)$ denotes the set of edges of the subgraph induced by the subset of vertices I.

The proof is also given in Appendix B. Note that Proposition 1 deals with the formula (7) and the linear program (12), which are well-defined for real (not necessarily integer) r_e.

Let us give an example of linear program (12) for the triangle network with weights from Fig. 2(a):

$$\begin{aligned} &Z = r_{(1,2)} + r_{(1,3)} + r_{(2,3)} - R_{\mathrm{CO}}, \\ &R_{\mathrm{CO}} = \min_{R_1,R_2,R_3} (R_1 + R_2 + R_3) \quad \text{such that} \\ &R_1 + R_2 \ge r_{(1,2)}, \\ &R_1 + R_3 \ge r_{(1,3)}, \\ &R_2 + R_3 \ge r_{(2,3)}, \\ &R_1 \ge 0,\ R_2 \ge 0,\ R_3 \ge 0. \end{aligned} \quad (13)$$

Its solution is

$$Z = \begin{cases} r_{(1,2)} + r_{(1,3)}, & \text{if } r_{(1,2)} + r_{(1,3)} \le r_{(2,3)}, \\ r_{(1,2)} + r_{(2,3)}, & \text{if } r_{(1,2)} + r_{(2,3)} \le r_{(1,3)}, \\ r_{(1,3)} + r_{(2,3)}, & \text{if } r_{(1,3)} + r_{(2,3)} \le r_{(1,2)}, \\ \frac{1}{2}[r_{(1,2)} + r_{(1,3)} + r_{(2,3)}], & \text{otherwise.} \end{cases} \quad (14)$$

That is, if there exists a bipartite rate that is larger than the sum of the two others, then the conference key rate is limited by the sum of the two smaller rates. Such bottleneck structures will be considered in more detail in Sec. V A. Otherwise, the conference key rate is one half of the sum of all bipartite rates. If all three bipartite rates are equal to one, then we recover the result $r_{\mathrm{conf}} = Z = 3/2$ we obtained before.

The linear program (12) results from Csiszár and Narayan's theorem applied to our case (a general formulation is given in Sec. B 1 in Appendix B). The first term in the right-hand side of Eq. (12a) is the total amount of (bipartite) randomness in the network generated per round. In the second term, R_i is the amound of bits announced by the ith participant per round such that each participant can recover the private randomness of all other participants ("omniscience"). Then R_{CO} denotes the minimal total amount of announced information for omniscience ("communication for omniscience"). Each constraint (12c) reflects the fact that the participants from $[N]\backslash I$ (if we consider them as a whole) are ignorant only about the bipartite bits that are "internal" for the subgraph induced by the vertices I. Intuitively, this constraint can be understood as follows: The total amount of information announced by the participants from I cannot be smaller than the total amount of randomness in their subnetwork. In summary, the maximally possible conference key rate is the total amount of randomness in the network minus the amount of publicly revealed information about this randomness.

Thus, Proposition 1 relates the linear program (12) originating from information-theoretic considerations with the graph-theoretic formulas (5) and (7).

V. FURTHER APPLICATIONS OF THE STP PROTOCOL

The main application of the proposed spanning-tree-packing protocol is the optimal conference key propagation in a network of bipartite QKD links. In this section, we discuss two further applications: upper bounds on the conference key propagation from graph topology and bottleneck structures (subsection V A), and network optimization, namely, optimal allocation of new bipartite QKD links in the network design (subsection V B).

A. Upper bounds on conference key propagation from graph topology: Complex bottleneck structures

Direct optimization using the Nash-Williams-Tutte formula (7) for calculating the conference key propagation rate requires exponential time in the number of parties N because the number of vertex partitions is exponential. There exists an efficient algorithm solving this problem [42], but it has no simple analytic expression. For practice it is often useful to have simple bounds for the quantity of interest. While a lower bound can be found by finding an arbitrary (generally, suboptimal) spanning tree packing, the Nash-Williams-Tutte formula gives upper bounds. Namely, it follows that

$$r_{\mathrm{conf}} \leq \frac{1}{|P|-1} \sum_{e \in \mathcal{E}(P)} r_e \tag{15}$$

for any vertex partition P. Consideration of the partition into single-element subsets (we will refer to it as the finest partition) $\mathcal{V}_i = \{i\}$, $i = 1, \dots, N$, gives

$$r_{\mathrm{conf}} \leq \frac{1}{N-1} \sum_{e \in \mathcal{E}} r_e. \tag{16}$$

That is, if we take an arbitrary vertex partition P, for example, the finest partition, then Ineq. (15) or (16) gives a simple upper bound. In the rest of this subsection, we discuss how to "guess" a vertex partition P which gives the tight bound or at least a good one, and consider examples depicted on Figs. 5 and 6.

Bound (15) also follows from simple arguments based on Eq. (5): k edge-disjoint spanning trees have $k(N-1)$ edges in total. This number cannot exceed the total number $n|\mathcal{E}| = n\sum_{e\in\mathcal{E}} r_e$ of edges in the multigraph $(\mathcal{V}, \mathcal{E}, \{nr_e\})$. A generalization of these arguments to an arbitrary partition P gives Ineq. (15).

From these arguments, it follows that, if the equality in Ineq. (16) holds, i.e.,

$$r_{\mathrm{conf}} = \frac{1}{N-1} \sum_{e \in \mathcal{E}} r_e, \tag{17}$$

then all edge capacities (bipartite QKD links) are fully used and there are no "leftover" edges. The first three rows of Fig. 4 give examples of this case. The bottom row in Fig 4 is an example of the inverse case. It is obvious that the conference key rate cannot exceed one because the conference key rate cannot exceed the total number of bipartite secret key bits of an arbitrary participant. The figure shows a decomposition of this graph into one spanning tree and a residual edge $(1,3)$. This corresponds to the case that the participants do not use the key $K_{(1,3)}$ in the generation of the conference key.

The existence of "leftover" QKD links so that Ineq. (16) is strict, or, in other words, if the minimum in the Nash-Williams-Tutte formula (7) is achieved not by the finest partition $\mathcal{V}_i = \{i\}$, $i = 1, \dots, N$, this indicates the existence of bottleneck structures in our network, which restrict the conference key rate. The minimum in the last example of Fig. 4 is achieved by the vertex bipartition into the subsets $\mathcal{V}_1 = \{1,2,3\}$ and $\mathcal{V}_2 = \{4\}$.

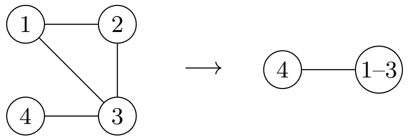

Figure 5. A contraction of the graph from Fig. 4 (last row) indicating a bottleneck structure, which restricts the conference key rate: connection of the node 4 to the rest of the network. The conference key rate in the original graph cannot be larger that the conference (bipartite) key rate in the contracted graph. If the link $(3,4)$ has the rate $r_{(3,4)} = 1$, then, for the left graph, $r_{\mathrm{conf}} \leq 1$.

Fig. 5 gives another interpretation of Ineq. (15). Instead of partitions, one can think about a *contraction* of the graph [40]. Namely, for a given partition $P = \{\mathcal{V}_\alpha\}$ consider the graph where the sets $\mathcal{V}_\alpha$ are vertices and two vertices $\mathcal{V}_\alpha$ and $\mathcal{V}_\beta$ are connected by an edge iff the set $\mathcal{E}(\mathcal{V}_\alpha, \mathcal{V}_\beta)$ of edges between the vertex subsets $\mathcal{V}_\alpha$ and $\mathcal{V}_\beta$ is nonempty. This edge obtains the weight

$$r_{\alpha\beta} = \sum_{e \in \mathcal{E}(\mathcal{V}_\alpha, \mathcal{V}_\beta)} r_e. \tag{18}$$

Then Ineq. (15) is intuitively clear. A contraction of a subset of vertices into one vertex means that the corresponding participants become actually one participant, i.e., they have common knowledge without cost of communication. Another interpretation: we can set the weights of edges (i.e., the bipartite secret key rates) connecting vertices from the same subset to infinity. This situation is more advantageous for conference key agreement, hence, the conference secret key rate cannot decrease under this transformation.

Generally, consider a bipartition $\mathcal{V} = \mathcal{V}_1 \sqcup \mathcal{V}_2$. Then, $|P| = 2$ and, according to formula (15), the conference key rate cannot exceed the total capacity of edges between the subsets $\mathcal{V}_1$ and $\mathcal{V}_2$. This is an obvious upper bound. The first three lines in the solution (14) for the triangle network depicted on Fig. 2(a) also reflect bottleneck structures based on bipartitions. E.g., if the "weakest" part of the network is the connection of vertex 1 to the rest of the network, then the contraction of vertices 2 and 3 gives the tight upper bound. Upper bounds based on bipartitions are used, e.g., in Ref. [21].

However, formula (15) can be used to obtain upper bounds based on more general partitions. An example is given in Fig 6, where the bottleneck structure is not a connection between two subgraphs, but a triangle. If, as in the previous examples, each edge corresponds to the bipartite key rate 1, then we obtain the upper bound $r_{\mathrm{conf}} \leq 3/2$ (actually, $r_{\mathrm{conf}} = 3/2$) originating from a triangle contracted graph, while considerations of only bipartitions give a loose bound $r_{\text{conf}} \leq 2$.

Finally, it is interesting to note that, in order to check that the minumum in Eq. (7) is achieved by the finest partition, i.e., Eq. (17) is satisfied, one does not need to check all possible partitions, but only partitions of the form $P_I = \{\mathcal{V}_1, \dots, \mathcal{V}_l, \mathcal{V}_{l+1}\}$, where $I = \{v_1, \dots, v_l\} \subsetneq [N]$, $\mathcal{V}_i = \{v_i\}$, $i = 1, \dots, l$, and $\mathcal{V}_{l+1} = [N] \backslash I$, see Corollary 2 in Appendix B. In other words, to ensure Eq. (17), it is sufficient to check that

$$\frac{1}{N-1} \sum_{e \in \mathcal{E}} r_e \leq \frac{1}{|I|} \sum_{e \in \mathcal{E}(I) \cup \mathcal{E}(I, [N] \backslash I)} r_e, \tag{19}$$

or, equivalently,

$$\frac{1}{N-|I|-1} \sum_{e \in \mathcal{E}([N] \backslash I)} r_e \leq \frac{1}{|I|} \sum_{e \in \mathcal{E}(I) \cup \mathcal{E}(I, [N] \backslash I)} r_e \tag{20}$$

for all $I \subsetneq [N]$. Here $\mathcal{E}(I, I') \subset E$ denotes the subset of edges connecting the vertices from I to the vertices from I'. As we show in Appendix B, for each I, Ineqs. (19) and (20) are either both satisfied or both violated.

Ineq. (20) has the following interpretation. The left-hand side of it is an upper bound on the conference key propagation in the subnetwork $[N] \backslash I$, see Eq. (15). Analogously, the right-hand side of Ineq. (20) is the upper bound on the conference key rate for the network where the vertices from $[N] \backslash I$ are contracted into one vertex (hence, the corresponding graph contains $|I|+1$ vertices). The violation of this inequality means that, even if we contract the set of vertices $[N] \backslash I$ into one vertex, the conference key rate in such simplified network is still upper bounded by the upper bound for the subnetwork $[N] \backslash I$. This indicates that the connections of vertices from I with each other and with the other vertices constitute a bottleneck structure.

Thus, the proved equivalence between the Nash-Williams–Tutte formula and the linear program (12) reveals new facts about the spanning-tree-packing problem.

B. Network optimization

Let us use the results from the previous subsection to optimize r_{conf} by adding bipartite links to networs in an optimal way. Consider the problem depicted in Fig 7 as an example.

Initially, we have a ring of six nodes [black solid edges in Fig 7(a)]. We assume $r_e = 1$ for all edges. Then the initial conference key rate is $r_{\text{conf}} = 6/5$. Note that the minimum in the Nash-Williams–Tutte formula (7) is achieved by the finest partition $\mathcal{V} = \{1\} \sqcup \ldots \sqcup \{6\}$ and, thus, Eq. (17) is satisfied.

Suppose that we are allowed to add one more bipartite QKD link, either $(1, 4)$ or $(2, 6)$ [red dashed edges in Fig 7(a)]. If we add the link $(1, 4)$, then the minimum in the Nash-Williams–Tutte formula (7) is still achieved by the finest partition and Eq. (17) is satisfied, which gives the new conference key rate $r_{\text{conf}} = 7/5$.

In contrast, if we choose to add the link $(2, 6)$ instead, then the minimum in the Nash-Williams–Tutte formula is achieved by the partition

$$\mathcal{V} = \{1, 2, 6\} \sqcup \{3\} \sqcup \{4\} \sqcup \{5\}, \tag{21}$$

which gives $r_{\text{conf}} = 4/3$. The conference key rate $4/3$ corresponds to a ring of four nodes ("4-ring"). Thus, the described contraction reveals the bottleneck structure in the form of a 4-ring if the link $(2, 6)$ is added. Adding the link (1,4) leads to a higher rate.

If we are allowed to add a second additional bipartite QKD link, e.g., $(2, 6)$, $(3, 6)$, or $(1, 5)$ depicted in Fig 7(b), then the choices $(2, 6)$ and $(3, 6)$ [i.e., the links that connect different parts of the graph separated by the central vertical link $(1, 4)$] lead to a further increase of the conference key rate to $r_{\text{conf}} = 8/5$. This again correspond to the finest partition as the optimal one and no bottleneck structures. In contrast, the choice $(1, 5)$ leads to the partition

$$\mathcal{V} = \{1, 4, 5, 6\} \sqcup \{2\} \sqcup \{3\} \tag{22}$$

as the optimal one, which corresponds to the contraction of nodes 1, 4, 5, and 6 and, thus, to a triangle bottleneck structure with $r_{\text{conf}} = 3/2$.

Thus, if we are allowed to allocate a restricted number of bipartite QKD links, bottleneck structures should be avoided.

VI. SECURITY PARAMETER OF THE STP PROTOCOL

So far, we assumed the case of perfectly secure bipartite keys. Let us now consider a realistic QKD scenario, where bipartite keys K_e, $e \in \mathcal{E}$, are characterized by the values $\varepsilon_e > 0$ of the security parameter, which is based on the trace distance between the real and ideal classical-quantum states after the protocol [54, 55]. Recall that $\varepsilon_e = 0$ corresponds to perfect security. Roughly speaking, ε_e can be associated with the probability that the distributed key is insecure, see Refs. [58, 59] for cryptographic operational interpretations.

Consider a tree network like in Fig 3. If the number of nodes in the network is N, then any tree contains $N - 1$ edges. As we know from Sec. III A, the conference key propagation protocol for the tree network can be described as a sequence of one-time-pad encryptions of a chosen bipartite key using the other bipartite keys. During such process, the security parameters of all keys are summed up [54, 55, 60], hence the security parameter of the conference key is

$$\varepsilon_{\text{conf}} = \sum_{e \in \mathcal{E}} \varepsilon_e. \tag{23}$$

If all ε_e are equal to ε, then $\varepsilon_{\text{conf}} = (N-1)\varepsilon$.

Consider now the general case and formula (5): a conference key is generated using k spanning trees T_β,

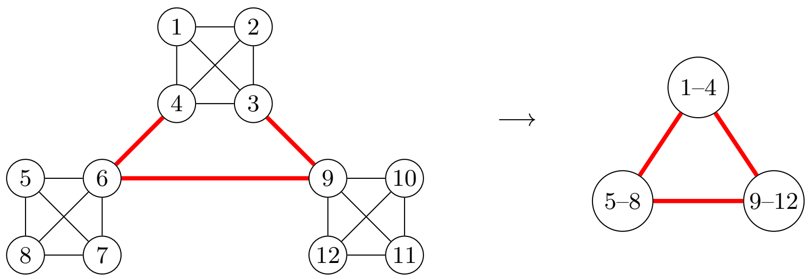

Figure 6. A contraction of a graph indicating a more complex bottleneck structure, than a connection between two subgraphs (like in Fig 5): a triangle bottleneck structure. If all bipartite QKD links have the rate $r_e = 1$ and, thus, the triangle network with bipartite key rates $r_e = 1$ has conference key rate $3/2$ (see Fig. 4), then, for the left graph, $r_{\mathrm{conf}} \leq 3/2$.

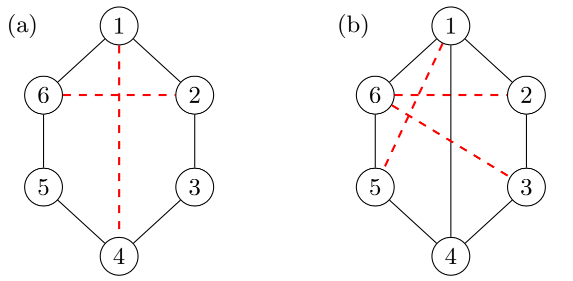

Figure 7. Network optimization. All edges here correspond to bipartite secret key rate $r = 1$. (a): The initial ring-like network of 6 nodes gives the conference key rate $r_{\mathrm{conf}} = 6/5$. If we are allowed to add one link (edge), then which of two links marked as red dashed edges is it better to choose? The optimal choice is to add the link $(1, 4)$. (b): If we are allowed to add one more link, which of three marked options is optimal? The optimal solution is to add $(2, 6)$ or $(3, 6)$ because the addition of the link $(1, 5)$ leads to a triangle bottleneck structure, see the text.

$\beta = 1, \ldots, k$. Then, if we form separate conference keys from each spanning tree, then their security parameters are

$$\varepsilon_{\mathrm{conf}}^{(\beta)} = \sum_{e \in T_\beta} \varepsilon_e. \tag{24}$$

If we merge all these spanning trees into one long conference key, then we need to sum all the security parameters given above:

$$\varepsilon = \sum_{\beta=1}^{k} \varepsilon_{\mathrm{conf}}^{(\beta)} = \sum_{\beta=1}^{k} \sum_{e \in T_\beta} \varepsilon_e. \tag{25}$$

If all ε_e are equal to the same number ε, then $\varepsilon_{\mathrm{conf}}^{(\beta)} = (N-1)\varepsilon$ and $\varepsilon_{\mathrm{conf}} = k(N-1)\varepsilon$. Thus, increasing k and n in Eq. (5) allows one to achieve a higher asymptotic conference key rate, but decreases its security parameter if we merge the keys obtained from all spanning trees into one long key.

VII. CONCLUSIONS AND OPEN PROBLEMS

We have proposed an optimal solution for the conference key agreement in an arbitrary network of bipartite QKD links. This solution is based on an optimal spanning tree packing (STP) in a graph of bipartite keys. The main tool for proving optimality is the proof of the equivalence between the graph-theoretic Nash-Williams–Tutte formula for the spanning-tree-packing number and a linear program originating from information-theoretic considerations of Csiszár and Narayan.

We have shown that the optimality of the STP protocol and the Nash-Williams–Tutte formula can be used for bounding the conference key rate from above and for optimization of QKD link allocations in the network design. Revealing bottlenecks of arbitrary topologies (i.e., originating from arbitrary vertex partitions), not just "linear" bottleneck structures (i.e., originating from bipartitions) play a crucial role here.

We highlight three open problems. First, we assumed that the bipartite key rates r_e are given. However, in practice, not all bipartite QKD links may operate at full capacities simultaneously due to a restricted number of QKD devices. This leads to different optimization problems, which include the optimization over the usage of the QKD links (i.e., over r_e) with given link capacities and node constraints. Such problems warrant further investigation.

The second open problem concerns the conference key generation in a subnetwork, where only a subset $\mathcal{V}' \subset \mathcal{V}$ of users want to establish a conference key, but other users are trusted and can assist them. Then, instead of spanning trees, we need to consider so called Steiner trees, i.e., trees that contain a given set of vertices (but also may include other vertices). Steiner tree packing for the relative problem of GHZ (Greenberger-Horne-Zeilinger) state distillation in a network of Bell pairs was considered in Refs. [61–63]. Unfortunately, in contrast to the spanning tree packing, the Steiner tree packing is an NP-hard problem. Nevertheless, does the optimal Steiner tree packing also provide optimal conference key agreement in a subnetwork among all all possible protocols?

Finally, it is interesting to generalize these results to the case where genuine multipartite QKD (e.g., based on GHZ states) [18] is available between some of the nodes. For example, there are not only bipartite, but also tripartite QKD links in the network. This leads to a transition from the concept of graphs to the concept of hypergraphs. The notions of spanning tree and the spanning-tree-packing are naturally generalized for hypergraphs. An open question is its optimality in this more general case.

Acknowledgments. We thank Federico Grasselli, Giacomo Carrara, and Justus Neumann for fruitful discussions. This work was funded by the German Federal Ministry of Education and Research (Project QuKuK, BMBF Grant No. 16KIS1618K).

Appendix A: A simple method of finding optimal spanning tree packing

Here we present a method of finding optimal spanning tree packing for integer edge weights $r_e = L_e$. Formally, this algorithm is not polynomial since it requires checking Ineq. (19) for all $I \subsetneq [N]$. Moreover, some steps of this protocol are defined heuristically, non-rigorously. Nevertheless, this simple method allowed us to find optimal spanning tree packings for the examples considered in this paper without addressing to the complicated algorithm from Ref. [42] and, probably, can be useful also for readers wishing to find spanning tree packings for small graphs or graphs with recognizable structures (certain patterns of edges, recognizable clusters, etc.).

Consider first the case of integer edge wedge $r_e = L_e$ and no bottlenecks in the network, i.e., Eq. (17) is satisifed, or, equivalently, Ineq. (19) is satisfied for all $I \subsetneq [N]$. Then we can generate $L_{\rm conf} = \sum_e L_e$ conference key bits in $N-1$ rounds.

The algorithm works then as follows.

Algorithm 1 Basic algorithm

```
Input: Graph with integer edge weights $L_e$ satisfying Eq. (17)
Output: Optimal spanning tree packing
1: $L_{\rm conf} \leftarrow \sum_e L_e$                    ▷ Conference key length
2: $L'_e \leftarrow (N-1)L_e$ for all edges $e$           ▷ New edge weights
3: for $\alpha \in \{1, \dots, L_{\rm conf}-2\}$ do
4:     Choose a spanning tree $T_\alpha$ with edges of maximal weights $L'_e$
5:     $L'_e \leftarrow L'_e - 1$ for all $e \in T_\alpha$
6: end for
7: Choose the last two spanning trees $T_{L_{\rm conf}-1}$ and $T_{L_{\rm conf}}$
```

Let us comment the (heuristic) rules of choice of the spanning trees. In each step, we need to choose a spanning tree with edges of possibly maximal weights L'_e. Of course, in some cases it is impossible, e.g., when there are $N-1$ edges of the maximal weight, but they do not form a spanning tree. However, generally, the algorithm prescribes to prefer edges with higher weights. The choices of the first $L_{\rm conf}-2$ trees can be arbitrary satisfying this rule. However, the choice of the last but one spanning tree is trickier because, after this choice and the corresponding reduction of the edge weights, the remaining edges of non-zero weight may not form a spanning tree. Hence, the choice of the spanning tree $T_{L_{\rm conf}-1}$ must ensure that the remaining graph is a spanning tree, which will be the last spanning tree spanning trees $T_{L_{\rm conf}}$. An example of the work of this algorithm is given in Fig. 8 (the first and second lines).

Consider now the case when Eq. (17) can be not satisfied, or, equivalently, Ineqs. (19) can be violated. Then the method is to break the graph into subgraphs until these conditions are satisfied and Algorithm 1 can be applied. Let us rewrite condition (19) for an arbitrary graph $(\mathcal{V}', \mathcal{E}')$ and integer weights L_e:

$$\frac{1}{|\mathcal{V}'|-1} \sum_{e \in \mathcal{E}'} L_e \leq \frac{1}{|I|} \sum_{e \in \mathcal{E}'(I) \cup \mathcal{E}'(I, \mathcal{V}' \setminus I)} L_e, \tag{A1}$$

where the notations $\mathcal{E}'(I)$ and $\mathcal{E}'(I, \mathcal{V}'\setminus I)$ for $I \subset \mathcal{V}'$ are defined analogously to $\mathcal{E}(I)$ and $\mathcal{E}(I, \mathcal{V}'\setminus I)$. We noticed [see the text after Ineq. (20)] that violation of Ineq. (A1) for some I means that the graph with $|I|+1$ vertices where vertices from $\mathcal{V}'\setminus I$ are contracted into one is a bottleneck structure. This observation suggests to consider two spanning tree packing problems separately: for the subgraph induced by the vertices $\mathcal{V}'\setminus I$ and for the aforementioned contracted graph, and then merge the spanning trees. Such splitting of the problem into two problems for smaller graphs works recursively as follows:

Algorithm 2 General algorithm

```
Input: Graph $(\mathcal{V}', \mathcal{E}')$ with integer edge weights $L_e$
Output: Optimal spanning tree packing
1: if Ineq. (A1) is satisfied for all $I \subsetneq \mathcal{V}'$ then
2:     Run Basic algorithm for the graph $(\mathcal{V}', \mathcal{E}')$
3: else if Ineq. (A1) is violated for some $I \subsetneq \mathcal{V}'$ then
4:     Run General algorithm for the graph $(\mathcal{V}'', \mathcal{E}'')$ obtained by the contraction of the vertices from $\mathcal{V}'\setminus I$ into one vertex
5:     Run General algorithm for the subgraph induced by the vertices $\mathcal{V}'\setminus I$
6:     Merge the spanning trees of the graphs from steps 4 and 5 according to the edge correspondence between the input graph $(\mathcal{V}', \mathcal{E}')$ and its contraction $(\mathcal{V}'', \mathcal{E}'')$
7: end if
```

An example of the work of this algorithm is given in Fig. 8 (the third line).

The necessity to check Ineqs. (19) for all vertex subsets I and, moreover, further checks Ineqs. (A1) for subgraphs, make this algorithm inefficient for large graphs. However, it can be applied if the graph contains well recognizable clusters so that finding I violating Ineqs. (19) and (A1) is obvious, or, vice versa, when we can guess that there are no such clusters and Eq. (17) is true so that

Basic algorithm can be directly applied. Otherwise, one can modify the algorithm to find suboptimal spanning tree packings.

So, we can hope that the algorithm can be applied to practical networks as well. There are well-known examples in computer science when formally non-polynomial algorithms (e.g., the simplex algorithm in linear programming) work well for most practical problems.

Finally, in the case of non-integer edge weights r_e, two ways can be considered. The first one is to consider n rounds such that all nr_e are integer. The second (heuristic) one is to solve the problem for all weights equal to one (or integers reproducing the proportions between r_e with some precision) and, with the obtained list of spanning trees, solve the linear program (B15) (see below an equivalent formulation of the optimal spanning tree packing problem) optimizing their weights. Note that Eq. (B15) is indeed a linear program for w_α if the set of the trees is fixed. The requirement that w_α must be rational is not a restriction since, if r_e are rational, there exists an optimal solution of the linear program with rational w_α.

Appendix B: Proof of optimality of the spanning-tree-packing conference key propagation

In this section we prove Theorem 1 and Proposition 1.

1. Reduction to a linear program for a multigraph network

The basic tool is the information-theoretic result by Cziszár and Narayan [56] about the classical conference secret key capacity. One of the models considered in their paper is as follows. In each repetition (round), independent and identically distributed (iid) N-tuples of random variables $(X_1, \dots, X_N)$ are delivered to the users. That is, the user i observes the random variable X_i. The random variables X_i for different i are generally dependent according to a known joint distribution for $(X_1, \dots, X_N)$, but the N-tuples for different rounds are independent. The participants communicate over a noiseless public channel. The eavesdropper has no information about X_i, but has access to the public communication. The participants want to agree on a common conference secret key with the highest possible rate. See Ref. [56] for formal definitions, but they match our definitions of the conference key capacity (10). Then the conference secret key capacity is given by

$$\begin{aligned}
&Z = H(X_{[N]}) - R_{\mathrm{CO}},\\
&R_{\mathrm{CO}} = \min_{R_1,\dots,R_N} \sum_{i=1}^{N} R_i \quad \text{such that}\\
&\sum_{i\in I} R_i \geq H(X_I | X_{[N]\setminus I}), \quad \forall I \subsetneq [N],
\end{aligned} \tag{B1}$$

where $X_I = (X_i)_{i\in I}$, H denotes the Shannon entropy, R_i is the amount of bits of information per round announced by the ith participant, and R_{CO} is the smallest achievable rate of "communication for omniscience", i.e., the minimal total amount of bits of public communication (per round) which allows all participants to recover all iid repetitions of all random variables $X_1, \dots, X_N$. The information-theoretic meaning of these relations and constraints is given after Theorem 1.

The definition of X_i for our case is described in Sec. IV. Each participant i obtains $\sum_{e\in\mathcal{E}_i} r_e$ bipartite perfectly secure bits per round, where $\mathcal{E}_i$ is the set of edges incident to the vertex i. This bits constitute X_i. Of course, different X_i are not independent since, for each edge $e = (i, j)$, r_e bits are included into both X_i and X_j. Thus,

$$\begin{aligned}
H(X_{[N]}) &= \sum_{e\in\mathcal{E}} r_e,\\
H(X_I | X_{[N]\setminus I}) &= \sum_{e\in\mathcal{E}(I)} r_e,
\end{aligned} \tag{B2}$$

from which we obtain the linear program (12). Thus, the proof of Theorem 1 has been reduced to Proposition 1, which we prove in the next subsection.

2. Proof of Proposition 1

Lemma 1. *Consider an arbitrary partition $P = \{\mathcal{V}_1, \dots, \mathcal{V}_p\}$ of the vertex set $\mathcal{V}$ into nonempty subsets, i.e., $\mathcal{V} = \mathcal{V}_1 \sqcup \ldots \sqcup \mathcal{V}_p$, $p \equiv |P| \geq 2$. Then the following upper bound for Z form linear program (12) holds:*

$$Z \leq \frac{1}{|P|-1} \sum_{e\in\mathcal{E}(P)} r_e, \tag{B3}$$

where, recall, $\mathcal{E}(P)$ is the set of cross-edges in the graph, i.e., the edges whose vertices lie in different partition subsets.

Proof. It will be convenient to introduce the notation

$$r[\mathcal{E}'] = \sum_{e\in\mathcal{E}'} r_e \tag{B4}$$

for an arbitrary subset $\mathcal{E}' \subset \mathcal{E}$. The sum of the constraints (12c) for $I_\alpha = [N]\setminus\mathcal{V}_\alpha$ for all $\alpha = 1, \dots, |P|$ gives

$$\begin{aligned}
(|P|-1)\sum_{i=1}^{N} R_i &\geq \sum_{\alpha=1}^{|P|} r[\mathcal{E}([N]\setminus\mathcal{V}_\alpha)]\\
&= \sum_{\alpha=1}^{|P|} \Big(r[\mathcal{E}] - r[\mathcal{E}(\mathcal{V}_\alpha)] - r[\mathcal{E}(\mathcal{V}_\alpha, [N]\setminus\mathcal{V}_\alpha)] \Big)\\
&= (|P|-1) r[\mathcal{E}] - r[\mathcal{E}(P)],
\end{aligned} \tag{B5}$$

where, recall, $\mathcal{E}(I, I') \subset \mathcal{E}$ denotes the subset of edges connecting the vertices from I to the vertices from I'.

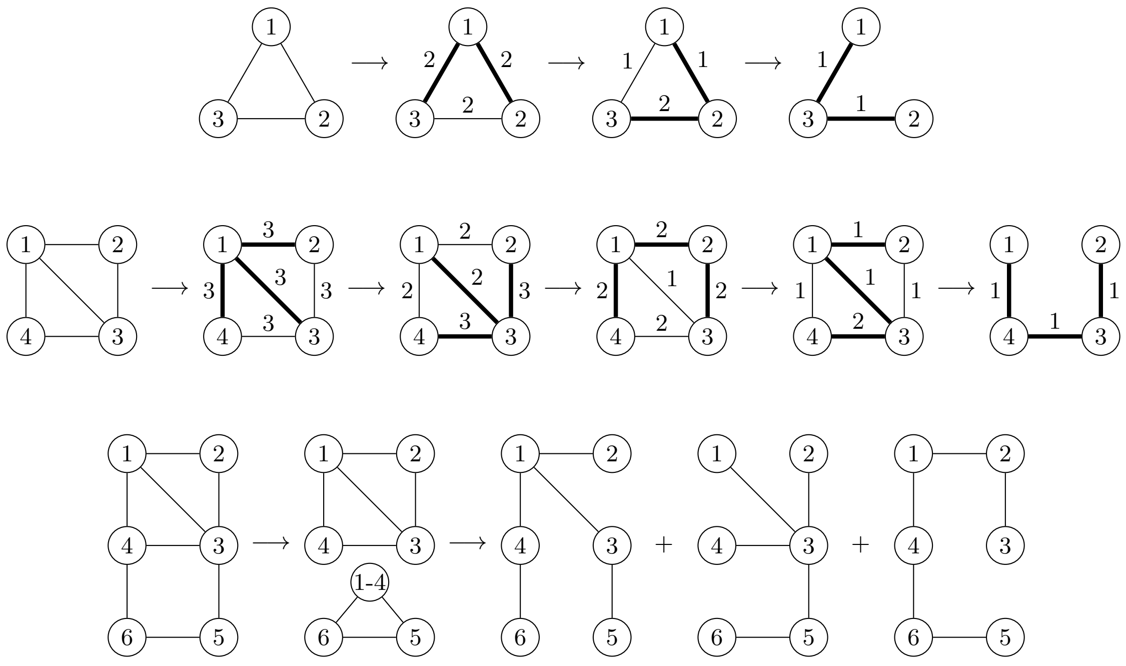

Figure 8. Illustration of the Basic algorithm (first two lines) and the General algorithm (the last line) of finding an optimal spanning tree packing. All edge weights are equal to one. For the graphs in the first two lines, condition (17) is satisfied and we know that we can generate certain number of conference key bits in $N-1$ round ($N-1$ is the number of vertices). We assign new weights $N-1$ to each edge and then, in each step, choose a spanning tree with possibly maximal weights of edges (bold edges). Each use of the edge reduce its weight by one. The graph in the third line violates Ineq. (19) for $I=\{5,6\}$, so, the conference key rate cannot exceed that for the graph where the rest vertices $\{1,2,3,4\}$ are contracted into one, i.e., the triangle graph, or 3/2. Then we solve the spanning-tree packing problem for both the contracted graph and the subgraph induced by the vertices $\{1,2,3,4\}$ and merge the spanning trees of these graphs according to the edge correspondence between the original graph and the contracted one. We obtained three spanning trees with not more than two uses of each edge, which gives the rate 3/2. Choices of other spanning trees lead to other optimal solutions.

Here we have used that each R_i participates in exactly $|P|-1$ constraints for various α,

$$\sum_\alpha r[\mathcal{E}(\mathcal{V}_\alpha,[N]\backslash\mathcal{V}_\alpha)] = 2r[\mathcal{E}(P)], \tag{B6}$$

(since each cross-edge is counted twice), and

$$\sum_\alpha r[\mathcal{E}(\mathcal{V}_\alpha)] + r[\mathcal{E}(P)] = r[\mathcal{E}]. \tag{B7}$$

Substitution of the lower bound for R_{CO} from Ineq. (B5) to Eq. (12a) gives Ineq. (B3). □

Lemma 1 and the tree packing (Nash-Williams–Tutte) theorem allows us to establish a relation between the optimal solution of the linear program and spanning tree packings in the multigraph.

Corollary 1. *The optimal solution of the linear program (12) gives the value*

$$Z = \min_{\substack{\text{vertex}\\ \text{partitions } P}} \frac{1}{|P|-1}\sum_{e\in\mathcal{E}(P)} r_e, \tag{B8}$$

which coincides with the Nash-Williams–Tutte formula (7).

Proof. By Lemma 1, Z cannot exceed the right-hand side of Eq. (B8). From the other side, the the right-hand side of Eq. (B8) is achievable by the spanning-tree-packing protocol precisely in view of the Nash-Williams–Tutte formula (7). Note that it is valid also for real r_e. Indeed, for n rounds, we can consider the multigraph $(\mathcal{V},\mathcal{E},\{\lfloor nr_e\rfloor\})$. Its spanning-tree-packing number $L^{(n)}_{\mathrm{conf}}$ is given by Eq. (6) with nr_e replaced by $\lfloor nr_e\rfloor$. The limit of $L^{(n)}_{\mathrm{conf}}/n$ as $n\to\infty$ gives again Eq. (7). Hence, the right-hand side of Eq. (B8) is also (asymptotically) achievable and, hence, optimal. □

This finishes the proof Proposition 1 and Theorem 1.

Remark 1. In the proof of Corollary 1, we implicitly assumed that the conference key propagation protocol based on the spanning tree packing problem (B15) gives a feasible solution to the linear program. It follows from the results of Csiszár and Narayan: Any feasible algorithm of the conference key generation from bipartite secret keys must satisfy the restrictions of linear program (12). However, it is instructive to show it explicitly, which will be done in the next subsection.

Finally, let us consider the case when the minimum in Nash-Williams–Tutte formula (7) is achieved by the finest partition $P_{\text{finest}} = \{\{1\}, \dots, \{N\}\}$, i.e., Eq. (17) is satisfied. According the Nash-Williams–Tutte formula, we need to prove that

$$\frac{r[\mathcal{E}]}{N-1} \leq \frac{r[\mathcal{E}(P)]}{|P|-1} \tag{B9}$$

for all partitions P. However, it turns out that it is sufficient to check Ineq. (B9) only for partitions of the form $P_I = \{\mathcal{V}_1, \dots, \mathcal{V}_l, \mathcal{V}_{l+1}\}$, where $I = \{v_1, \dots, v_l\} \subsetneq [N]$, $\mathcal{V}_i = \{v_i\}$, $i = 1, \dots, l$, and $\mathcal{V}_{l+1} = [N] \backslash I$. Application of Ineq. (B9) to this particular form of a partition gives

$$\frac{r[\mathcal{E}]}{N-1} \leq \frac{r[\mathcal{E}(I)] + r[\mathcal{E}(I, [N]\backslash I)]}{|I|}. \tag{B10}$$

We can prove the following:

Proposition 2. *Ineq. (B9) is true for all vertex partitions P iff Ineq. (B9) is true for all partitions P_I of the form given above (i.e., for all nonempty $I \subsetneq [N]$).*

Proof. In one direction, the statement is obvious: Since Ineq. (B9) is a particular case of Ineq. (B9), then, if Ineq. (B9) is satisfied for all vertex partitions P, then Ineq. (B9) is satisfied for all partitions P_I. Let us prove the other direction.

Put by definition

$$R_i = r[\mathcal{E}_i] - \frac{r[\mathcal{E}]}{N-1}. \tag{B11}$$

Substitution of Eq. (B11) into Ineq. (12c) for an arbitrary I gives

$$2r[\mathcal{E}(I)] + r[\mathcal{E}(I, [N]\backslash I)] - \frac{|I|}{N-1} r[\mathcal{E}] \geq r[\mathcal{E}(I)], \tag{B12}$$

from which Ineq. (B10) follows. Hence, if all Ineq. (B10), we have an explicit solution satisfying all constraints (B9) and, as it follows from the proved results and also can be checked explicitly, gives $Z = r[\mathcal{E}]/(N-1)$.

□

Note that Ineqs. (B10) are trivially satisfied as equalities for $I = [N]\backslash\{k\}$, $k = 1, \dots, N$. Actually, Eq. (B11) is a solutions of the the system of N linear equations obtained by replacement of "$\geq$" by "$=$" in constraints (12c) for $I = [N]\backslash\{k\}$, $k = 1, \dots, N$. We obtained an explicit form of R_i yielding the optimal value for the linear program if there are no bottleneck structures in the network.

In the main text, we also use the equivalent form of Ineq. (B10):

$$\frac{r[\mathcal{E}([N]\backslash I)]}{N - |I| - 1} \leq \frac{r[\mathcal{E}(I)] + r[\mathcal{E}(I, [N]\backslash I)]}{|I|}. \tag{B13}$$

Indeed, Ineq. (B10) can be rewritten as

$$\frac{r[\mathcal{E}(I)] + r[\mathcal{E}(I, [N]\backslash I)] + r[\mathcal{E}([N]\backslash I)]}{N-1} \leq \frac{r[\mathcal{E}(I)] + r[\mathcal{E}([N]\backslash I)]}{|I|}, \tag{B14}$$

from which Ineq. (B13) follows.

3. Spanning-tree-packing protocol satisfies the information-theoretic constraints

Let us show that the conference key rate given by the Nash-Williams–Tutte formula (7) does not exceed C from (12), i.e., the information-theoretic constraints (12c) are satisfied. In principle, we do not need to prove this because it is a direct consequence of Csiszár and Narayan's result: The rate of any conference key propagation algorithm cannot exceed the information-theoretic bound (12). However, it is instructive to give a direct and constructive proof.

We will use another (equivalent) formulation of the spanning tree packing problem [42]. For a given graph $(\mathcal{V}, \mathcal{E})$ and "edge capacities" (in our case – bipartite key rates) $\{r_e\}_{e \in \mathcal{E}}$, we need to choose a finite set of spanning trees $\{T_\alpha\}$ of this graph and assign rational weights $\{w_\alpha \geq 0\}$ to them such that

$$\begin{aligned} &\sum_\alpha w_\alpha = r_{\text{conf}} \to \max_{\{w_\alpha\}}, \\ &\sum_{\alpha:\, T_\alpha \ni e} w_\alpha \leq r_e, \quad \text{for all } e \in \mathcal{E}, \end{aligned} \tag{B15}$$

where $T_\alpha \ni e$ means that the spanning tree T_α contains the edge e. The constraints in Eq. (B15) mean that each bipartite secret bit from each r_e can be used only in one spanning tree. In other words, if we generate one conference bit using the tree T_α, we spend one bit from each of the edges constituting this tree. The weight w_α is the average number of usages of the tree T_α per round. For example, the spanning tree packings depicted in the first line of Fig. 4, correspond to $\alpha \in \{1, 2, 3\}$ (indices of three spanning trees) and $w_\alpha = 1/2$ for each α: Each spanning tree is used once per two rounds. In the second and the third lines of Fig. 4, we have $w_\alpha = 1/3$ for each of five spanning trees and $w_\alpha = 1$ for each of two spanning trees, respectively. The constraints in Eq. (B15) mean that, for each edge, the total number of usages of the conference bits cannot exceed its "capacity" (number of bipartite bits generated per round) r_e.

Since, as we mentioned in Sec. III C, we use a slightly different formulation of the spanning tree packing problem in comparison with the standard one, let us prove the equivalence of Eqs. (5) and (B15).

Observation 3. *The optimization problems (5) and (B15) are equivalent.*

Proof. Indeed, from one side, consider a set of spanning trees $\{\widetilde{T}_\beta\}_{\beta=1}^k$ in the multigraph $(\mathcal{V}, \mathcal{E}, \{nr_e\})$. We can obviously map them into a set of spanning trees in the graph $(\mathcal{V}, \mathcal{E})$ by just ignoring the edge multiplicities. However, different spanning trees in the multigraph can correspond to the same spanning tree in the graph $(\mathcal{V}, \mathcal{E})$, see Fig. 9. Denote this map as f. Thus, f is in general a many-to-one correspondence. Denote $f^{-1}(T_\alpha)$ the preimage of T_α, i.e., the set of the spanning trees from

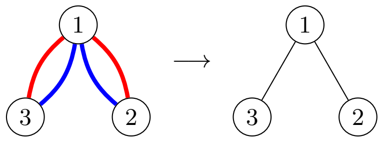

Figure 9. If we ignore the edge multiplicities of a multigraph, several spanning trees of it (here: two spanning trees of the multigraph, which are depicted by red and blue) can map into the same spanning tree of the resulting graph.

the multigraph $(\mathcal{V},\mathcal{E},\{nr_e\})$ mapped into the spanning tree T_α of the graph $(\mathcal{V},\mathcal{E})$. Then, put

$$w_\alpha = \frac{f^{-1}(T_\alpha)}{n} \tag{B16}$$

and

$$\sum_\alpha w_\alpha = \frac{k}{n}. \tag{B17}$$

Thus, a feasible solution of problem (5) is a feasible solution of problem (B15) with the same value of the conference key rate.

From the other side, a feasible solution $\{T_\alpha, w_\alpha\}$ of the problem (B15) with rational w_α can be mapped to a feasible solution of problem (5) in the same manner. Namely, there exists n such that all nw_α are integer. Then, by construction, we have found $k = n\sum_\alpha w_\alpha$ edge-disjoint spanning trees in the multigraph $(\mathcal{V},\mathcal{E},\{nr_e\})$.

Hence, problems (5) and (B15) are equivalent. □

We are ready to prove the main result of this subsection.

Lemma 2. *The conference key rate $r_{\rm conf}$ given by (5) can be expressed as*

$$r_{\rm conf} = \sum_{e\in\mathcal{E}} r_e - \sum_{i=1}^N R_i, \tag{B18}$$

where R_i satisfy constraints (12c).

According to the Csiszár-Narayan construction, R_i have the meaning of the amount of information sent by the ith participant (per round) to the public channel in the spanning tree packing algorithm. The proof is based on explicit calculation of the information announced by the participants. We will give the corresponding explanations in the proof, though the proof of a formal statement of the lemma does not rely on this interpretation of R_i.

Proof. Denote $d_i^{(\alpha)}$ the nodal degree of the vertex i in the αth spanning tree. According to the algorithm, for the αth spanning tree, the participant i, announce $R_i^{(\alpha)} = d_i^{(\alpha)} - 1$ bits of information. As we discussed after Eq. (B15), the weight w_α of the αth spanning tree is a fraction of using this spanning tree, i.e., for large number n of rounds, the αth spanning tree is used approximately $w_\alpha n$ times. So, the participant i announces on average $\sum_\alpha w_\alpha R_i^{(\alpha)}$ bits per round.

In addition, as it was mentioned in Sec. V A, not always the full capacity r_e of all edges is used (like in the bottom line of Fig. 4). Thus, the differences

$$\sum_{e\ni i}\Bigl(r_e - \sum_{\alpha:\,T_\alpha\ni e} w_\alpha\Bigr) \tag{B19}$$

for all edges e can be non-zero. In other words, in general, there are constraints in the spanning tree packing problem (B15) satisfied as strict inequalities.

Constraints in (12) assume "omniscience": Participants must be able to recover all bipartite secret bits. In this paradigm, the participants have to announce the bipartite secret bits not used in the conference key propagation protocol. Each "leftover" bipartite secret bit is shared by two participants. Let us demand that, on average, one half of this bit is announced by one participant and one half is announced by the other one. That is, in one half rounds this bit is announced by one participant and in one half of rounds – by the other one. Then, the total amount of information announced by the ith participant per round is

$$R_i = \sum_\alpha w_\alpha(d_i^{(\alpha)} - 1) + \frac{1}{2}\sum_{e\ni i}\Bigl(r_e - \sum_{\alpha:\,T_\alpha\ni e} w_\alpha\Bigr). \tag{B20}$$

We have defined R_i and now start the proof that, with such R_i, Eq. (B18) and constraints (12c) are satisfied. The proof will not use the interpretation of R_i as the amount of information announced by the participants.

For the right-hand side of Eq. (B18), we have

$$\begin{aligned}\sum_{e\in\mathcal{E}} r_e - \sum_{i=1}^N R_i &= \sum_{e\in\mathcal{E}} r_e - \sum_{i=1}^N\sum_\alpha w_\alpha(d_i^{(\alpha)} - 1)\\ &\quad - \sum_{e\in\mathcal{E}}\Bigl(r_e - \sum_{\alpha:\,T_\alpha\ni e} w_\alpha\Bigr)\\ &= \sum_\alpha w_\alpha\left(\sum_{e\in T_\alpha} 1 + N - \sum_{i=1}^N d_i^{(\alpha)}\right).\end{aligned} \tag{B21}$$

Each spanning tree has $N-1$ edges, hence $\sum_{e\in T_\alpha} 1 = N-1$ for all α. Also, the sum of nodal degreees of vertices in a subgraph is the number of edges in this subgraph multiplied by two (since each edge is counted twice), hence $\sum_i d_i^{(\alpha)} = 2(N-1)$ and

$$\begin{aligned}\sum_{e\in\mathcal{E}} r_e - \sum_{i=1}^N R_i &= \Bigl(\sum_\alpha w_\alpha\Bigr)[N-1+N-2(N-1)]\\ &= \sum_\alpha w_\alpha = r_{\rm conf}.\end{aligned} \tag{B22}$$

We have proved Eq. (B18)

Now we need to prove that R_i given by Eqs. (B20) satisfy constraints (12c). The substitution gives for an arbitrary nonempty $I \subsetneq [N]$:

$$\sum_{i\in I} R_i = \sum_\alpha w_\alpha \sum_{i\in I}(d_i^{(\alpha)}-1) + \frac{1}{2}\sum_{i\in I}\sum_{e\ni i}\Bigl(r_e - \sum_{\alpha:\, T_\alpha\ni e} w_\alpha\Bigr). \tag{B23}$$

Now, for each spanning tree, we have

$$\sum_{i\in I} d_i^{(\alpha)} = 2|\mathcal{E}_\alpha(I)| + |\mathcal{E}_\alpha(I,[N]\backslash I)|, \tag{B24}$$

where $\mathcal{E}_\alpha(I)$ is the set of edges of the subgraph of the αth spanning tree induced by vertices I and $\mathcal{E}_\alpha(I,[N]\backslash I)$ is the set of edges in the αth spanning tree connecting vertices from I to vertices outside I.

For the last sum in Eq. (B23), we have

$$\begin{aligned}\frac{1}{2}\sum_{i\in I}\sum_{e\ni i}\Bigl(r_e &- \sum_{\alpha:\, T_\alpha\ni e} w_\alpha\Bigr)\\ &= \sum_{e\in\mathcal{E}(I)}\Bigl(r_e - \sum_{\alpha:\, T_\alpha\ni e} w_\alpha\Bigr)\\ &+ \frac{1}{2}\sum_{e\in\mathcal{E}(I,[N]\backslash I)}\Bigl(r_e - \sum_{\alpha:\, T_\alpha\ni e} w_\alpha\Bigr)\\ &\geq r[\mathcal{E}(I)] - \sum_\alpha w_\alpha|\mathcal{E}_\alpha(I)|. \end{aligned}\tag{B25}$$

Substitution of Eq. (B24) and Ineq. (B25) into Eq. (B23) gives

$$\sum_{i\in I} R_i \geq r[\mathcal{E}(I)] + \sum_\alpha w_\alpha\Bigl[|\mathcal{E}_\alpha(I)| + |\mathcal{E}_\alpha(I,[N]\backslash I)| - |I|\Bigr]. \tag{B26}$$

To finish the proof of Ineqs. (12c), it is sufficient to prove that

$$|\mathcal{E}_\alpha(I)| + |\mathcal{E}_\alpha(I,[N]\backslash I)| \geq |I|. \tag{B27}$$

The left-hand side is the number of edges incident on the vertices from I in the αth spanning tree. Consider the subgraph $(I, \mathcal{E}_\alpha(I))$ of the αth spanning tree induced by the vertices I. It is not necessarily connected: Some vertices from I can be connected via vertices outside I.

For the clarity of arguments, consider first the case that the subgraph $(I, \mathcal{E}_\alpha(I))$ is connected. Then it is also a tree and thus contains $|I|-1$ edges. Also the set $\mathcal{E}_\alpha(I,[N]\backslash I)$ has at least one element, which connects the vertices from I to the vertices outside I (since each spanning tree is a connected graph). Hence, (B27) is true in this case.

Consider now the general case where the subgraph $(I, \mathcal{E}_\alpha(I))$ is not necessarily connected. Denote $\{I_\mu\}$ the subsets of I forming connected components of this subgraph. Then, again, each connected component contains is a tree and contains $|I_\mu|-1$ edges. Also, for each μ, there exists at least one edge of the αth spanning tree connecting the vertices from I_μ to the vertices outside I_μ (and outside I because otherwise such vertices would belong to the same connected component). Thus,

$$|\mathcal{E}_\alpha(I)| + |\mathcal{E}_\alpha(I,[N]\backslash I)| \geq \sum_\mu[(|I_\mu|-1)+1] = |I|. \tag{B28}$$

This finish the proof of Ineqs. (B27), (12c) and thus the whole lemma. $\square$

[1] C. H. Bennett and G. Brassard, Quantum cryptography: public key distribution and coin tossing, in *Proc. IEEE Int. Conf. Computers, Systems and Signal Processing* (Institute of Electrical and Electronics Engineers, New York, 1984) pp. 175–179.

[2] N. Gisin, G. Ribordy, W. Tittel, and H. Zbinden, Quantum cryptography, Rev. Mod. Phys. **74**, 145 (2002).

[3] V. Scarani, H. Bechmann-Pasquinucci, N. J. Cerf, M. Dušek, N. Lütkenhaus, and M. Peev, The security of practical quantum key distribution, Rev. Mod. Phys. **81**, 1301 (2009).

[4] S. Pirandola, U. 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, Advances in quantum cryptography, Adv. Opt. Photon. **12**, 1012 (2020).

[5] A. K. Fedorov, N. Gisin, S. M. Beloussov, and A. I. Lvovsky, Quantum computing at the quantum advantage threshold: a down-to-business review (2022), arXiv:2203.17181 [quant-ph].

[6] M. Mohseni, A. Scherer, K. G. Johnson, O. Wertheim, M. Otten, N. A. Aadit, K. M. Bresniker, K. Y. Camsari, B. Chapman, S. Chatterjee, G. A. Dagnew, A. Esposito, F. Fahim, M. Fiorentino, A. Khalid, X. Kong, B. Kulchytskyy, R. Li, P. A. Lott, I. L. Markov, R. F. McDermott, G. Pedretti, A. Gajjar, A. Silva, J. Sorebo, P. Spentzouris, Z. Steiner, B. Torosov, D. Venturelli, R. J. Visser, Z. Webb, X. Zhan, Y. Cohen, P. Ronagh, A. Ho, R. G. Beausoleil, and J. M. Martinis, How to build a quantum supercomputer: Scaling challenges and opportunities (2024), arXiv:2411.10406 [quant-ph].

[7] M. Epping, H. Kampermann, and D. Bruß, Large-scale quantum networks based on graphs, New J. Phys. **18**, 053036 (2016).

[8] M. Epping, H. Kampermann, and D. Bruß, Robust entanglement distribution via quantum network coding, New J. Phys. **18**, 103052 (2016).

[9] J. Wallnöfer, A. Pirker, M. Zwerger, and W. Dür, Multipartite state generation in quantum networks with optimal scaling, Sci. Rep. **9**, 314 (2019).

[10] S. De Bone, R. Ouyang, K. Goodenough, and D. Elk-

ouss, Protocols for creating and distilling multipartite GHZ states with Bell pairs, IEEE Trans. Quantum Eng. **1**, 4102710 (2020).

[11] P. Contreras-Tejada, C. Palazuelos, and J. de Vicente, Genuine multipartite nonlocality is intrinsic to quantum networks, Phys. Rev. Lett. **126**, 040501 (2021).

[12] D. Sukachev, Large quantum networks, Physics-Uspekhi **64**, 1021 (2021).

[13] G. Avis, F. Rozpędek, and S. Wehner, Analysis of multipartite entanglement distribution using a central quantum-network node, Phys. Rev. A **107**, 012609 (2023).

[14] M. Ghaderibaneh, H. Gupta, and C. Ramakrishnan, Generation and distribution of GHZ states in quantum networks, in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE Computer Society, Los Alamitos, CA, USA, 2023) pp. 1120–1131.

[15] E. Sutcliffe and A. Beghelli, Multiuser entanglement distribution in quantum networks using multipath routing, IEEE Trans. Quantum Eng. **4**, 1 (2023).

[16] H. Han, B. Liu, B. Tang, S. Xiong, J. Huang, W. Yu, and S. Chen, Differentiated service entanglement routing for quantum networks (2024), arXiv:2401.17503 [quant-ph].

[17] G. Vardoyan, E. van Milligen, S. Guha, S. Wehner, and D. Towsley, On the bipartite entanglement capacity of quantum networks, IEEE Trans. Quantum Eng. **5**, 4100414 (2024).

[18] G. Murta, F. Grasselli, H. Kampermann, and D. Bruß, Quantum conference key agreement: A review, Adv. Quantum Technol. **3**, 2000025 (2020).

[19] M. Epping, H. Kampermann, C. Macchiavello, and D. Bruß, Multi-partite entanglement can speed up quantum key distribution in networks, New. J. Phys. **19**, 093012 (2017).

[20] F. Salek and A. Winter, New protocols for conference key and multipartite entanglement distillation, IEEE Trans. Inf. Theory **71**, 4374 (2025).

[21] S. Pirandola, General upper bound for conferencing keys in arbitrary quantum networks, IET Quantum Commun. **1**, 22 (2020).

[22] S. Das, S. Bäuml, M. Winczewski, and K. Horodecki, General upper bound for conferencing keys in arbitrary quantum networks, Phys. Rev. X **11**, 041016 (2021).

[23] G. Carrara, H. Kampermann, D. Bruß, and G. Murta, Genuine multipartite entanglement is not a precondition for secure conference key agreement, Phys. Rev. Res. **3**, 013264 (2021).

[24] M. Proietti, J. Ho, F. Grasselli, P. Barrow, M. Malik, and A. Fedrizzi, Experimental quantum conference key agreement, Science Advances **7**, eabe0395 (2021).

[25] A. nag Oruganti, Multi-user QKD using quotient graph states derived from continuous-variable dual-rail cluster states (2024), arXiv:2412.14317 [quant-ph].

[26] H. Kimble, The quantum internet, Nature **453**, 1023 (2008).

[27] S. Pirandola and S. Braunstein, Physics: Unite to build a quantum Internet, Nature **532**, 169 (2016).

[28] C. Simon, Towards a global quantum network, Nature Photon. **11**, 678 (2017).

[29] S. Wehner, D. Elkouss, and R. Hanson, Quantum internet: A vision for the road ahead, Nature Photon. **11**, eaam9288 (2018).

[30] P. P. Rohde, *The Quantum Internet. The second Quantum Revolution* (Cambridge University Press, Cambridge, 2021).

[31] P. P. Rohde, Z. Huang, Y. Ouyang, H.-L. Huang, Z.-E. Su, S. Devitt, R. Ramakrishnan, A. Mantri, S.-H. Tan, N. Liu, S. Harrison, C. Radhakrishnan, G. K. Brennen, B. Q. Baragiola, J. P. Dowling, T. Byrnes, and W. J. Munro, The quantum Internet (technical version) (2025), arXiv:2501.12107 [quant-ph].

[32] N. H. Valencia, A. Ma, S. Goel, S. Leedumrongwatthanakun, F. Graffitti, A. Fedrizzi, W. McCutcheon, and M. Malik, A large-scale reconfigurable multiplexed quantum photonic network (2025), arXiv:2501.07272 [quant-ph].

[33] V. Kumar, C. Cicconetti, M. Conti, and A. Passarella, Quantum Internet: Technologies, protocols, and research challenges (2025), arXiv:2502.01653 [quant-ph].

[34] Y. Cao, Y. Zhao, Q. Wang, J. Zhang, S. X. Ng, and L. Hanzo, The evolution of quantum key distribution networks: On the road to the Qinternet, IEEE Commun. **24**, 839 (2021).

[35] A. Gaidash, G. Miroshnichenko, and A. Kozubov, Quantum network security dependent on the connection density between trusted nodes, J. Opt. Commun. Netw. **14**, 934 (2022).

[36] E. O. Kiktenko, A. Tayduganov, and A. K. Fedorov, Routing algorithm within the multiple non-overlapping paths' approach for quantum key distribution networks, Entropy **26**, 1102 (2024).

[37] Y. Piétri, P.-E. Verdier, B. Lacour, M. Gautier, H. Huang, T. Camus, J.-S. Pegon, M. Zuber, J.-C. Faugère, M. Schiavon, A. Rhouni, Y. Jaouën, N. Fabre, R. Alléaume, T. Rivera, and E. Diamanti, Quantum key distribution with efficient post-quantum cryptography-secured trusted node on a quantum network (2025), arXiv:2504.01454 [quant-ph].

[38] P. Horoschenkoff, J. Henrich, R. Böhn, I. Khan, J. Rödiger, M. Gunkel, M. Bauch, J. Benda, P. Bläcker, E. Eichhammer, U. Eismann, G. Frenck, H. Griesser, W. Jontofsohn, N. Kopshoff, S. Röhrich, F. Seidl, N. Schark, E. Sollner, D. von Blanckenburg, A. Heinemann, M. Stiemerling, and M. Gärtner, Demoquandt: A carrier-grade QKD network (2025), arXiv:2503.21186 [quant-ph].

[39] L. Mariani, R. Yehia, C. Pascual-García, F. Centrone, J. van der Kolk, M. Ángeles Serrano, and A. Acín, Quantum key distribution over complex networks (2025), arXiv:2504.02372 [quant-ph].

[40] R. Diestel, *Graph Theory*, 5th ed. (Springer, Berlin, 2017).

[41] E. Palmer, On the spanning tree packing number of a graph: a survey, Discrete Math. **230**, 13 (2001).

[42] F. Barahona, Packing spanning trees, Math. Oper. Res. **20**, 104 (1995).

[43] Y. Du, Y. Liu, C. Yang, X. Zheng, S. Zhu, and X.-s. Ma, Experimental measurement-device-independent quantum cryptographic conferencing, Phys. Rev. Lett. **134**, 040802 (2025).

[44] F. Grasselli, H. Kampermann, and D. Bruß, Conference key agreement with single-photon interference, New J. Phys. **21**, 123002 (2019).

[45] C. H. Park, M. K. Woo, B. K. Park, Y.-S. Kim, H. Baek, S.-W. Lee, H.-T. Lim, S.-W. Jeon, H. Jung, S. Kim, and S.-W. Han, $2 \times N$ twin-field quantum key distribution

network configuration based on polarization, wavelength, and time division multiplexing, npj Quantum Inf. **8**, 48 (2022).

[46] X. Zhong, W. Wang, R. Mandil, H.-K. Lo, and L. Qian, Simple multiuser twin-field quantum key distribution network, Phys. Rev. Appl. **17**, 014025 (2022).

[47] G. Carrara, G. Murta, and F. Grasselli, Overcoming fundamental bounds on quantum conference key agreement, Phys. Rev. Appl. **19**, 064017 (2023).

[48] J. Li, W. Wang, and H. F. Chau, Fully passive quantum conference key agreement (2024), arXiv:2407.15761 [quant-ph].

[49] S. Zhao, P. Zeng, W.-F. Cao, X.-Y. Xu, Y.-Z. Zhen, X. Ma, L. Li, N.-L. Liu, and K. Chen, Phase-matching quantum cryptographic conferencing, Phys. Rev. Appl. **14**, 024010 (2020).

[50] N. Stefanakos, G. Maragkopoulos, A. Mandilara, and D. Syvridis, A measurement device independent quantum key distribution protocol in the service of three users (2025), arXiv:2504.06902 [quant-ph].

[51] C. Huang, R. Guan, X. Liu, S. Li, W. He, H. Liang, Z. Luo, Z. Zhang, W. Li, and K. Wei, Fully connected twin-field quantum key distribution network (2025), arXiv:2504.15137 [quant-ph].

[52] H. Dong, C. Jiang, D. Ma, C. Zhang, J. Huang, H. Li, L.-X. You, Y. Liu, X.-B. Wang, Q. Zhang, and J.-W. Pan, Experimental multi-dimensional side-channel-secure quantum key distribution (2025), arXiv:2504.19242 [quant-ph].

[53] F. Kanitschar and C. Pacher, Security of multi-user quantum key distribution with discrete-modulated continuous-variables (2024), arXiv:2406.14610 [quant-ph].

[54] C. Portmann and R. Renner, Cryptographic security of quantum key distribution (2014), arXiv:1409.3525 [quant-ph].

[55] C. Portmann and R. Renner, Security in quantum cryptography, Rev. Mod. Phys. **94**, 025008 (2022).

[56] I. Csiszár and P. Narayan, Secrecy capacities for multiple terminals, IEEE Trans. Inf. Theory **50**, 3047 (2004).

[57] E. Chitambar, D. Leung, L. Mančinska, M. Ozols, and A. Winter, Everything you always wanted to know about locc (but were afraid to ask), Comm. Math. Phys. **328**, 303 (2014).

[58] I. Arbekov and S. Molotkov, Distinguishability of quantum states and Shannon complexity in quantum cryptography, J. Exp. Theor. Phys. **125**, 50 (2017).

[59] A. Trushechkin, On the operational meaning and practical aspects of using the security parameter in quantum key distribution, Quantum Electron. **50**, 426 (2020).

[60] M. Ben-Or, M. Horodecki, D. W. Leung, D. Mayers, and J. Oppenheim, The universal composable security of quantum key distribution, in *Theory of Cryptography*, edited by J. Kilian (Springer Berlin Heidelberg, Berlin, Heidelberg, 2005) pp. 386–406.

[61] S. Bäuml and K. Azuma, Fundamental limitation on quantum broadcast networks, Quantum Sci. Technol. **2**, 024004 (2017).

[62] S. Bäuml, K. Azuma, G. Kato, and E. D., Linear programs for entanglement and key distribution in the quantum internet, Comm. Phys. **3**, 55 (2020).

[63] K. Azuma, S. Bäuml, T. Coopmans, E. D., and B. Li, Tools for quantum network design, AVS Quantum Sci. **3**, 014101 (2021).