New Upper Bounds for Evolving Secret Sharing via Infinite Branching Programs

Bar Alon*
alonbar08@gmail.com

Amos Beimel*
amos.beimel@gmail.com

Tamar Ben David†
tamaryahav12@gmail.com

Eran Omri†
omrier@ariel.ac.il

Anat Paskin-Cherniavsky†
anps83@gmail.com

March 10, 2024

Abstract

Evolving secret-sharing schemes, defined by Komargodski, Naor, and Yogev [TCC 2016B, IEEE Trans. on Info. Theory 2018], are secret-sharing schemes in which there is no a-priory bound on the number of parties. In such schemes, parties arrive one by one; when a party arrives, the dealer gives it a share and cannot update this share in later stages. The requirement is that some predefined sets (called authorized sets) should be able to reconstruct the secret, while other sets should learn no information on the secret. The collection of authorized sets that can reconstruct the secret is called an evolving access structure. The challenge of the dealer is to be able to give short shares to the the current parties without knowing how many parties will arrive in the future. The requirement that the dealer cannot update shares is designed to prevent expensive updates.

Komargodski et al. constructed an evolving secret-sharing scheme for every monotone evolving access structure; the share size of the t^{th} party in this scheme is 2^{t-1}. Recently, Mazor [ITC 2023] proved that evolving secret-sharing schemes require exponentially-long shares for some evolving access structure, namely shares of size $2^{t-o(t)}$. In light of these results, our goal is to construct evolving secret-sharing schemes with non-trivial share size for wide classes of evolving access structures; e.g., schemes with share size 2^{ct} for $c < 1$ or even polynomial size. We provide several results achieving this goal:

- We define layered infinite branching programs representing evolving access structures, show how to transform them into generalized infinite decision trees, and show how to construct evolving secret-sharing schemes for generalized infinite decision trees. Combining these steps, we get a secret-sharing scheme realizing the evolving access structure.

  As an application of this framework, we construct an evolving secret-sharing scheme with non-trivial share size for access structures that can be represented by layered infinite branching programs with width at layer t of at most $2^{0.15t}$. If the width is polynomial, then we get an evolving secret-sharing scheme with quasi-polynomial share size.

*Department of Computer Science, Ben Gurion University of the Negev.
†Department of Computer Science, Ariel University, Ariel Cyber Innovation Center (ACIC).

- We construct efficient evolving secret-sharing schemes for dynamic-threshold access structures with high dynamic-threshold and for infinite 2 slice and 3-slice access structures. The share size of the t^{th} party in these schemes is $2^{\tilde{O}((\log t)^{1/\sqrt{2}+\varepsilon})}$ for any constant $\varepsilon > 0$, which is comparable to the best-known share size of $2^{\tilde{O}((\log t)^{1/2})}$ for finite 2-slice and 3-slice access structures.
- We prove lower bounds on the share size of evolving secret-sharing schemes for infinite k-hypergraph access structures and for infinite directed st-connectivity access structures. As a by-product of the lower bounds, we provide the first non-trivial lower bound for finite directed st-connectivity access structures for general secret-sharing schemes.

Contents

1 Introduction

In the common model of secret-sharing schemes [15, 39, 28] there are n parties and a dealer, which holds a secret. The dealer applies some randomized algorithm to the secret, resulting in n strings, called shares; it gives the i^{th} share to the i^{th} party. There are two requirements. (1) correctness: some predefined subsets of the parties can jointly reconstruct the secret from their shares, and (2) security: any other set gets no information on the secret. The collection of predefined authorized sets is called an access structure. These schemes are well-studied and have many applications. This model assumes that the number of parties is known when preparing the shares and giving the shares to the parties; furthermore, the sharing algorithm and the share size are determined by the number of parties, e.g. in the best-known secret-sharing scheme for an arbitrary n-party access structure the share size is $2^{0.585n}$ [5]. The assumption that the number of parties is known in advance is problematic in many scenarios. Of course, one can take some upper bound on the number of parties. On one hand, if this bound is big, then the share size will be large even if only few parties actually participate in the scheme. On the other hand, if this bound is small, then there is a risk that too many parties will arrive and no further shares can be produced; this will require an expensive re-sharing of the secret and updating all shares (which can be impossible if some parties are temporally off-line). Thus, we need to consider models with an unbounded number of parties.

Komargodski, Naor, and Yogev [30] defined *evolving secret-sharing schemes* with an unbounded number of parties. In this model, parties arrive one after the other and the number of parties that will arrive is not known. In the beginning of the execution, the dealer holds a secret (as in the common model). When a party arrives, the dealer computes a share and gives it to the party; this share cannot be updated in the future. Thus, when preparing the t^{th} share the dealer cannot assume any bound on the number of parties that will eventually arrive; the size of the t^{th} share should be measured as a function of t. We require correctness and privacy with respect to an *evolving access structure*, where the parties are $\{p_i\}_{i\in\mathbb{N}}$ and the evolving access structure is a collection of finite subsets of the parties that are authorized to reconstruct the secret.[1]

We next briefly discuss the known results on evolving secret-sharing schemes. A longer discussion can be found in Section 1.2. Komargodski et al. [30] showed that every monotone evolving access structure can be realized by an evolving secret-sharing scheme; in this scheme the size of the t^{th} share is 2^{t-1}. Recently, Mazor [35] proved that evolving secret-sharing schemes require exponentially long shares – there is an evolving access structure such that in any evolving secret-sharing scheme realizing it the size of share of the t^{th} party is $2^{t-o(t)}$ (for infinitely many t's). On the positive side, Komargodski et al. and follow-up works [31, 23, 9, 24, 10, 18, 22, 36, 37, 41, 42] constructed efficient evolving secret-sharing schemes for natural access structures.

1.1 Our Results

Our first goal in this research is to give constructions of evolving secret-sharing schemes with non-trivial share size for wide classes of evolving access structures; e.g., schemes with share size 2^{ct} for $c < 1$ or sub-exponential share size. Our second goal is to construct efficient evolving secret-sharing schemes, i.e., schemes with polynomial-size shares, for natural classes of evolving access structures. Finally, our third goal is to characterize the exact share size required in evolving secret-sharing

[1] We assume that the order that the parties arrive is known in advance, or, alternatively, the t^{th} party to arrive assumes the role of the t^{th} party.

schemes realizing interesting access structures. We provide several results achieving these goals. See Table 1 for a summary of our results.

Evolving Secret-Sharing Schemes for Infinite Branching Programs. We abstract and generalize the constructions of [30, 31] of evolving secret-sharing schemes. We define infinite branching programs, which represent evolving access structures, show how to transform them to generalized infinite decision trees, and show how to construct evolving secret-sharing schemes for generalized infinite decision trees (abbreviated GIDT). We deffer the discussion on GIDTs to Section 1.3. Thus, to construct an evolving secret-sharing scheme for an evolving access structure using our framework, one can either represent it as an infinite branching program and use our transformation to construct a generalized infinite decision tree or directly represent the evolving access structure as a generalized infinite decision tree. We note that many secret-sharing schemes for finite access structures use a representation of the access structure by some computational model to construct a secret-sharing scheme realizing the access structure, e.g., CNF and DNF formulas are used in [28], monotone formulas are used in [14], and monotone span programs are used in [29].

An infinite monotone non-deterministic branching program (abbreviated IBP) computes a monotone function $f : \{0,1\}^* \to \{0,1\}$; this function is the characteristic function of an evolving access structure.[2] An IBP is an infinite directed acyclic graph G, where each edge is labeled by a variable from $\{x_i\}_{i\in\mathbb{N}}$ or by the constant 1. For every input $\sigma \in \{0,1\}^t$ (interpreted as an assignment to the variables $x_1, \ldots, x_t$) it holds that $f(\sigma) = 1$ if and only if there exists a path in G from the source vertex to a leaf (a vertex without out-going edges) that is satisfied by σ, that is, each edge in the path is either labeled by 1 or labeled with a variable x_i such that $1 \leq i \leq t$ and $\sigma_i = 1$ (see Definition 3.1 for a formal definition). A layered IBP (abbreviated LIBP) is an IBP, where the vertices are partitioned into finite layers, all edges are directed from some layer i to layer $i+1$, and all edges entering layer i are labeled by either 1 or by x_i. See Figure 1 for an illustration of an LIBP. Intuitively, when using LIBPs for constructing evolving secret-sharing scheme, passing through an edge that is labeled by x_i is interpreted as using the share of the i^{th} party in the reconstruction.

We show how to reduce the question of realizing an evolving access structure represented as an LIBP to the question of realizing certain finite access structures. One parameter that determines the share size of the resulting evolving secret-sharing scheme is the width of the LIBP, where the width of an LIBP at layer t, denoted by $w(t)$, is the number of vertices in layer t. We prove the following theorem.

Theorem 1.1 (Realizing LIBPs – Informal)**.** *Let B be an LIBP. There exists an evolving secret-sharing scheme realizing B in which the share of party p_t is the shares of p_t in $\left(\prod_{1\leq j\leq \log t} w(2^j)\right)$ secret-sharing schemes realizing some finite access structures.*

As an application of this framework, we construct evolving secret-sharing schemes for LIBPs with bounded width.

Theorem 1.2 (Realizing Bounded Width LIBPs – Informal)**.** *For every function $\varepsilon(t) < 0.04$, every LIBP of width $w(t) \leq 2^{\varepsilon(t)\cdot t}$ can be realized by an evolving secret-sharing scheme in which the share size of the t^{th} party is*

$$2^{O\left(\min\left\{\varepsilon(t)\log t, \sqrt{\varepsilon(t)}\right\}\right)\cdot t}.$$

[2] That is, $f(\sigma_1, \ldots, \sigma_t) = 1$ if an only if $\{p_i : 1 \leq i \leq t, \sigma_i = 1\}$ is in the access structure.

In the above theorem, $\varepsilon(t)$ can be an arbitrary function that is smaller than 1. For example, for LIBPs whose width is at most $2^{0.15t}$ (i.e., $\varepsilon(t) = 0.15$), we get share size $2^{0.97t}$. As another example, if the width is polynomial (i.e., $\varepsilon(t) = O(\log(t)/t)$), then we get an evolving secret-sharing scheme with quasi-polynomial share size. Thus, evolving access structures that can be represented by LIBPs with bounded width can be realized with non-trivial share size. This is in contrast with the lower bound of [35], proving that there exists an evolving access structure requiring shares of size at least $2^{t-o(t)}$.

Efficient Evolving Secret-Sharing Schemes for dynamic-threshold for Large Thresholds. In an evolving $\mathrm{tr}(\cdot)$-dynamic-threshold access structure, where $\mathrm{tr} : \mathbb{N} \to \mathbb{N}$ is a function, a set of parties A is authorized if for some $t \in \mathbb{N}$, the set A contains at least $\mathrm{tr}(t)$ parties from the first t parties. Komargodski and Paskin-Cherniavsky [31] constructed an evolving $\mathrm{tr}(\cdot)$-dynamic-threshold secret-sharing scheme in which the share size of the t^{th} party is $\tilde{O}(t^4)$. We show how to construct a more efficient evolving secret-sharing scheme when the dynamic-threshold function is large, i.e., $\mathrm{tr}(t) \geq t - t^\beta$ for some constant $0 < \beta < 1$.

Theorem 1.3 ($(t - t^\beta)$-Dynamic-Threshold Secret-Sharing Schemes – Informal)**.** *Let $\beta \in (0, 1)$ be a constant and $\mathrm{tr}(t) \geq t - t^\beta$. There exists an evolving secret-sharing scheme realizing the evolving $\mathrm{tr}(\cdot)$-dynamic-threshold access structure in which the share size of the t^{th} party is at most $\tilde{O}(t^{1+2\sqrt{\beta}+\beta})$.*

For all $\beta < 1$, our scheme is more efficient than the scheme of [31]. For example, for $\mathrm{tr}(t) = t - t^{1/4}$, the share size in our scheme is $\tilde{O}(t^{2.25})$ (compared to $\tilde{O}(t^4)$ in the scheme of [31]).

Efficient Evolving Secret-Sharing Schemes for Slice Access Structures. An infinite k-slice access structure is an access structure where all sets of size at most $k - 1$ are unauthorized, all finite sets of size at least $k + 1$ are authorized, and each set of size k can be either authorized or unauthorized; that is, to specify a k-slice access structure we need to specify which sets of size k (i.e., in the k^{th}-slice) are authorized. Secret sharing for finite slice access structures have been extensively studied (see, for example, the citations in [2]); they are equivalent to conditional disclosure of secrets (CDS) protocols, a cryptographic primitive introduced by Gertner et al. [27]. 2-slice access structures are also known as forbidden-graph secret-sharing schemes [40]. We construct efficient evolving secret-sharing schemes for infinite 2-slice access structures and 3-slice access structures.

Theorem 1.4 (Realizing 2-Slice and 3-slice Access Structures – Informal)**.** *Every 2-slice and 3-slice access structure can be realized by an evolving secret-sharing scheme in which the share size of the t^{th} party is $2^{\tilde{O}\left((\log t)^{\varepsilon+1/\sqrt{2}}\right)}$, for any constant $\varepsilon > 0$.*

The share size in the best-known secret-sharing schemes for finite n-party k-slice access structures for constant k is $2^{\tilde{O}(k+\sqrt{k \log n})} = 2^{\tilde{O}(\sqrt{\log n})}$ (using the CDS protocols of [33, 34] and the transformation of [7, 1]). For infinite 2-slice and infinite 3-slice access structures, the share size in our evolving secret-sharing scheme is comparable.

Lower Bounds. We prove lower bounds on the share size of evolving secret-sharing schemes for two natural classes of access structures. We first consider infinite directed st-connectivity access structures; in these access structures the parties are edges of an infinite graph (with some order

on the edges determining when they arrive in the evolving access structure); a set of parties (i.e., edges) is authorized if and only if it contains a path from a fixed source vertex to a fixed target vertex. We obtain the following lower bound.

Theorem 1.5 (Lower Bounds for Evolving Directed st-Connectivity – Informal)**.** *There exists an evolving directed st-connectivity access structure, such that in every evolving secret-sharing scheme realizing it the total share size of the t^{th} party, for infinitely many t's, is at least $\Omega(t)$.*

As a by-product of the lower bounds, we provide the first non-trivial lower bound for finite directed st-connectivity access structures for general secret-sharing schemes.

Theorem 1.6 (Lower Bounds for Finite Directed st-connectivity – Informal)**.** *For every $n \in \mathbb{N}$ there exists an n-party directed st-connectivity access structure, such that in every secret-sharing scheme realizing it, there exists at least one party whose share size is at least $\Omega(\sqrt{n})$.*

Previously, no non-trivial lower bound was known for finite st-connectivity access structures for general secret-sharing schemes. A lower bound of $n^{\Omega(\log n)}$ for linear secret-sharing schemes was proven by Pitassi and Robere [38].

We also prove lower bounds on the share size of evolving secret-sharing schemes for infinite k-hypergraph access structures for a constant k; in these access structures the minimal authorized sets are of size exactly k; however, there can be large unauthorized sets. Our lower bounds for infinite k-hypergraph access structures for constant k are tight as a fairly naive evolving secret-sharing scheme provides a matching upper bound on the share size.

Theorem 1.7 (Lower Bounds for Evolving k-Hypergraphs – Informal)**.** *For every constant k there exists an evolving k-hypergraph access structure, such that in every evolving secret-sharing scheme realizing it, the share size of the t^{th} party, for infinitely many t's is at least $\Omega(t^{k-2})$.*

1.2 Previous Results

1.2.1 Evolving Secret-Sharing Schemes

We first mention two related works that preceded [30]. Cachin [17] and Csirmaz and Tardos [21] studied online secret sharing, which is similar to evolving secret-sharing schemes. As in evolving secret-sharing schemes, in online secret-sharing, parties arrive one after the other and the number of parties is unbounded. However, in [21] the number of authorized sets that a party can join is bounded and in [17] there is a large public bulletin board.

Evolving Threshold Secret-Sharing Schemes. Komargodski et al. [30] constructed an evolving k-threshold secret-sharing schemes for any constant k in which the size of the share of the t^{th} party is $O(k \log t)$. D'Arco, De Prisco, and De Santis [22] constructed an improved evolving 3-threshold secret-sharing scheme in which the size of the share of the t^{th} party is $(4/3 + \varepsilon) \log t$ for arbitrary small ε (the share size in the evolving 3-threshold scheme of [30] is at least $2 \log t$). D'Arco et al. [23] constructed probabilistic evolving k-threshold secret-sharing schemes in which the share size is $O(1)$; in these schemes the secret is reconstructed only with a constant probability $p < 1$. Other constructions of evolving threshold secret-sharing schemes were given in [24, 36, 42].

	Upper bounds	Lower bounds
General evolving access structures	2^{t-1} [30]	$\frac{2^{t-o(t)}}{t}$ [35]
LIBPs with width $\leq 2^{\varepsilon(t)\cdot t}$	$2^{O(\min\{(\sqrt{\varepsilon(t)},\varepsilon(t)\log t\}\cdot t)}$ Thms. 3.15, 3.17	
Evolving k-hypergraphs	$O(t^{k-1})$ Theorem A.1	$\Omega(t^{k-2})$ Theorem 6.8
Evolving 2, 3-slices	$2^{\tilde{O}\left((\log t)^{1/\sqrt{2}+\varepsilon}\right)}$ for any constant $\varepsilon > 0$ Thms. 5.4, 5.6	
$(t - t^{\beta})$-dynamic threshold for a constant $\beta \in (0, 1)$.	$\tilde{O}(t^{1+2\sqrt{\beta}+\beta})$ Thm. 4.1	
Evolving directed st-connectivity	$t^{O(\log t)}$ (layered graphs) Theorem 3.19	$\Omega(t)$ Theorem 6.7

Table 1: A summary of the known lower and upper bounds on the share size in evolving secret-sharing schemes for the evolving access structures considered in this paper.

Evolving Dynamic-Threshold Secret-Sharing Schemes. Komargodski and Paskin-Cherniavsky [31] constructed an evolving $\mathrm{tr}(\cdot)$-dynamic-threshold secret-sharing scheme in which the share size of the t^{th} party is $O(t^4 \log t)$. Xing and Yuan [41] showed an alternative construction of evolving $\mathrm{tr}(\cdot)$-dynamic secret sharing scheme. Their construction saves a factor of $\log t$ compared to the evolving scheme of a [31]. They also considered the evolving $\mathrm{tr}(t) = t^\beta$-dynamic-threshold secret-sharing schemes (that is, the dynamic-threshold is small) and showed that it can be realized by an evolving secrets-sharing scheme in which the share size of the t^{th} party is $O(t^{4\beta})$. We show that this can be achieved (up to a factor of $\log t$) by a variation of the scheme of [31]. In this paper we use the construction of [41] to construct evolving $\mathrm{tr}(t)$-dynamic-threshold secret-sharing schemes for large dynamic-threshold $\mathrm{tr}(\cdot) \geq t - t^\beta$ for some constant β.

Evolving Secret-Sharing Schemes for Other Access Structures. Chaudhury, Dutta, and Sakurai [18] constructed evolving threshold schemes that can be implemented in the complexity class AC^0. Dutta, Roy, Fukushima, Kiyomoto, and Sakurai [25] and Phalakarn, Suppakitpaisarn, Attrapadung, and Matsuura [37] constructed evolving hierarchical secret-sharing schemes (in the latter paper the schemes are homomorphic). Beimel and Othman [9, 10] constructed evolving ramp secret-sharing schemes, i.e., schemes in which there is a gap between the dynamic threshold $\mathrm{tr}_2(\cdot)$ for authorized sets and the dynamic-threshold $\mathrm{tr}_1(\cdot)$ for unauthorized sets. They showed that for every constants $0 < \alpha < \beta < 1$, there is an evolving $(\mathrm{tr}_1(t) = \alpha \cdot t, \mathrm{tr}_2(t) = \beta \cdot t)$-ramp secret-sharing scheme in which the size of the shares of each party is $O(1)$.

1.2.2 Some Related Works on Secret-Sharing Schemes for Finite Access Structures

Secret-sharing schemes for arbitrary access structures were introduced by Ito, Saito, and Nishiseki [28]; they constructed for every monotone n-party access structure a secret-sharing scheme in which the size of the share of each party is 2^n. In a breakthrough work, Liu, and Vaikuntanathan [32] constructed a secret-sharing scheme for arbitrary access structures with share size $2^{0.944n}$. This was improved in a sequence of works [34, 2, 3, 5]; currently, the best known secret-sharing schemes for arbitrary access structures were constructed by Applebaum and Nir [5] and have share size $2^{0.585n}$. The best known lower bound on the share size is by Csirmaz [20, 19], proving that for every $n \in \mathbb{N}$ there is an n-party access structure in which the share size of at least one party is $\Omega(n/\log n)$ and its total share size is at least $\Omega(n^2/\log n)$.

We next mention some results for finite counterparts of the evolving access structures considered in this paper. Finite *undirected* st-connectivity access structures can be realized by a secret-sharing scheme in which each share and the secret is a bit [13]. The best known secret sharing scheme for finite *directed* st-connectivity access structures is by using the formula based-secret-sharing scheme of [14] and has share size $n^{O(\log n)}$ for realizing a graph with n edges. This scheme can be used to realize an access structure represented as a finite non-deterministic branching program of size n with share size $n^{O(\log n)}$. The best constructions for k-slice access structures are by various transformations from the k-server CDS protocols of [33, 34]; the best schemes known today have share size $\min\{2^{O(k)+\tilde{O}(\sqrt{k \log n})}, kn \cdot 2^{\tilde{O}(\sqrt{k \log n})}, 2^{\tilde{O}(\sqrt{n})}\}$ [1, 2, 8]. The naive secret-sharing scheme for k-hypergraph access structures is to share the secret independently for each minimal authorized set, this results in share size $O(\binom{n}{k-1})$ per party. This can be improved by a factor of $\log n$ using a result of Erdös and Pyber [26]. Recently, it was proved by Beimel [6] that for every n and every $3 \leq k \leq \log n$, there is a k-hypergraph with n vertices in which the share size of at least one party is $\Omega(n^{1-1/(k-1)}/k)$ and its total share size is at least $\Omega(n^{2-1/(k-1)}/k)$.

1.3 Our Techniques

Layered Infinite Branching Programs and Generalized Infinite Decisions Trees. Our task is to design an evolving secret-sharing scheme for LIBPs. We do not know how to construct such a scheme directly. We know how to realize LIBPs when the infinite graph of the infinite branching program is a tree, using the evolving secret-sharing scheme of [30] for undirected st-connectivity. However, transforming a (layered) graph of a LIBP B to a tree results in a tree whose width is huge – for every path $u_0, u_{j_1}, \ldots, u_{j_t}$ from the source vertex to a vertex in the t^{th}-layer there is a vertex $u_{j_1,\ldots,j_t}$ in the t^{th} layer of the tree.

Following [30, 31], we overcome this problem by partitioning the variables of the layered branching program into consecutive sets, called generations. The generations are defined by some function $h : \mathbb{N} \to \mathbb{N}$; the i^{th} generation contains the variables $x_{h(i-1)-1}, \ldots, x_{h(i)}$. In the infinite decision tree T we construct, there is a vertex $u_{j_1,\ldots,j_i}$ for any sequence of vertices $u_{j_1}, \ldots, u_{j_i}$ in layers $h(1), \ldots, h(i)$ respectively. If the width of the LIBP B in layer t (i.e., the number of vertices in the layer) is $w(t)$, then the number of vertices in the resulting infinite branching program T is $O\left(\prod_{1\leq j\leq i} w(h(j))\right)$ (this is the expression in Theorem 1.1, taking $h(i) = 2^i$). We add an edge $(u_{j_1,\ldots,j_{i-1}}, u_{j_1,\ldots,j_{i-1},j_i})$ to T representing all paths in B from $u_{j_{i-1}}$ in layer $h(i-1)$ to u_{j_i} in layer $h(i)$; this edge should be satisfied by an assignment σ if and only if σ satisfies some path in B from $u_{j_{i-1}}$ to u_{j_i}.

We abstract the above construction by defining a generalized infinite decisions tree (abbreviated GIDT), which is an infinite tree together with a partition function $h : \mathbb{N} \to \mathbb{N}$; a GIDT is an infinite tree in which each edge between layer $i-1$ and layer i in the tree is labeled by a predicate that depends on the variables in the i^{th} generation, i.e., on $x_{h(i-1)-1}, \ldots, x_{h(i)}$. To construct an evolving secret-sharing scheme for a GIDT, we first execute the evolving secret-sharing scheme of [30] for the tree; in this scheme the parties are the edges of the tree. Next, for each edge in the tree we take the share sh_e of the edge and share it using a secret-sharing realizing the predicate of the edge (here, again, we represent an access structure by a predicate).

Evolving Secret-Sharing Schemes for LIBPs with Bounded Width. The main application of our construction of evolving secret-sharing schemes for LIBPs is an evolving secret-sharing scheme with non-trivial share size realizing LIBPs with bounded width. In Theorems 3.15 and 3.17, we present two constructions for LIBPs with bounded width. Both constructions use the transformation from LIBPs to GIDTs and use the evolving secret-sharing scheme realizing the GIDT, that is, we use Theorem 1.1. For example, by Theorem 1.1, if the width of the LIBP is $w(t) = 2^{0.04t}$, then the number of shares of secret-sharing schemes for finite access structures that the t^{th} party holds is

$$O\left(\prod_{1\leq j\leq \log t} w(2^j)\right) = O\left(\prod_{1\leq j\leq \log t} 2^{0.04\cdot 2^j}\right) = O\left(2^{0.04\cdot\sum_{1\leq j\leq \log t} 2^j}\right)$$
$$\leq O\left(2^{0.04\cdot 2^{\log(t+1)}}\right) = O\left(2^{0.04(t+1)}\right).$$

Thus, the number of shares for width $w(t) = 2^{0.04t}$ is non-trivial. We still need to specify how we realize the finite access structures determined by the labels of the GIDT. In the first construction, we use the best-known secret-sharing scheme for arbitrary n-party access structures of Applebaum

and Nir [5]; the share size in this scheme is $2^{0.585n}$. In the second construction, we use the formula-based secret-sharing scheme of Benaloh and Leichter [14] using a monotone formula for the graph reachability problem. When the width of the LIBP is smaller than $2^{t/\log^2 t}$, the second construction is more efficient.

Evolving Secret-Sharing Schemes for Evolving dynamic-threshold Access Structure with Large Threshold. We use the evolving secret-sharing scheme of Xing and Yuan [41] for dynamic-threshold access structures. In this scheme, the parties are partitioned into generations. In each generation, with parties $\{p_i, p_{i+1}, \ldots, p_{i+g}\}$, two schemes are executed: (1) a secret-sharing realizing the finite restriction of the evolving $\mathrm{tr}(\cdot)$-dynamic-threshold access structure to the parties of the generation, and (2) Shamir's $\mathrm{tr}(g)$-out-of-$(g + \mathrm{tr}(g))$ threshold secret-sharing scheme. Each party in the generation gets a share of each scheme and the last $\mathrm{tr}(g)$ shares of Shamir's scheme are recursively shared using the evolving scheme of Xing and Yuan among the next generations.

We improve the share size for large dynamic-threshold access structures, i.e., when $\mathrm{tr}(t) \geq t - t^\beta$ for some constant $0 < \beta < 1$, by constructing a better secret-sharing scheme for the finite $(t - t^\beta)$-dynamic-threshold access structure. Specifically, we consider the access structure whose parties are $\{p_1, \ldots, p_g\}$ and a set A is authorized if $|A \cap \{p_1, \ldots, p_t\}| \geq \mathrm{tr}(t) = t - t^\beta$ for some $1 \leq t \leq g$. This access structure can be realized by executing g copies of Shamir's secret-sharing scheme, i.e., for each $1 \leq t \leq g$ we execute Shamir's $\mathrm{tr}(t)$-out-of-t secret-sharing scheme. We prove that for large $\mathrm{tr}(\cdot)$ it suffices to execute only t^β copies of Shamir's scheme. Assume that $\mathrm{tr}(t) \geq \mathrm{tr}(t-1) + 1$ and consider an authorized set A whose maximum party is p_t; if $|A \cap \{p_1, \ldots, p_t\}| \geq \mathrm{tr}(t)$, then

$$|A \cap \{p_1, \ldots, p_{t-1}\}| \geq |A \cap \{p_1, \ldots, p_t\}| - 1 \geq \mathrm{tr}(t) - 1 \geq \mathrm{tr}(t-1).$$

Thus, if $\mathrm{tr}(t) \geq \mathrm{tr}(t-1) + 1$ we do not need to execute the $\mathrm{tr}(t)$-out-of-t secret-sharing scheme. We show that this leaves us with at most t^β schemes.

Evolving Secret-Sharing Scheme for Evolving Slice Access Structures. We next explain the ideas of our construction of an evolving secret-sharing scheme for a 2-slice access structure, in which the authorized sets are some sets of size two and all sets of size at least 3. First, we handle authorized sets of size at least 3 using the scheme of Komargodski et al. [30]. For authorized sets of size exactly 2 we do the following. Partition the parties into generations. Let G_i denote the i^{th} generation, and let k be a large constant. We then use the secret-sharing scheme for finite slice functions [33, 7] to share s among the parties of every k consecutive generations. Finally, we need to handle pairs of parties in the access structure that are not in k consecutive generations; here for every $j \in \mathbb{N}$ we give the j^{th} party, which is in some generation G_i, a random bit r_j. Then, for every t in generation at least $i + k$, if the j^{th} and t^{th} parties are in the access structure, we give $s \oplus r_j$ to the t^{th} party. The size of the share of the t^{th} party in some generation i is dominated by the share in the secret-sharing scheme for the finite slice functions and the number of bits $s \oplus r_j$ that it gets; the latter number is at most the number of parties in the first $i - k$ generations. By choosing the correct size of the generations (namely $2^{\log^c i}$ for some constant c), we get the desired share size. By considering arbitrarily large k, we show that the share size decreases.

The evolving secret-sharing scheme for a 3-slice access structure uses similar ideas; however, it is more complicated. Specifically, the complicated case in constructing an evolving secret-sharing scheme for 3-slice access structures is in the case where there are two parties in some generation

and one party from a future generation. To handle this case we use a CDS protocol for the finite index function. The details of this scheme are given in Section 5.3.

Lower Bounds for Evolving Secret Sharing of Some Natural Access Structures. We prove lower bounds on the share size for explicit natural evolving access structures. Toward proving these results, we first show a general lower bound. This lower bound generalizes the recent result of [35] to include more access structures, and is inspired by the generalization of Csirmaz's lower bound [20] due to Blundo et al. [16]. The idea is to define an infinite independent sequence: we partition the parties into two sets $A = \{p_{a_i}\}_{i\in\mathbb{N}}$ and $B = \{p_{b_i}\}_{i\in\mathbb{N}}$ and consider an infinite sequence of sets $A_1, A_2, \ldots$, each of them is a finite subset of A, and consider an evolving access structure whose minimal authorized sets are $\{A_i \cup \{b_i\}\}_{i\in\mathbb{N}}$ (the definition of an infinite independent sequence is more general; see Definition 6.4). Using the lower bound of [20, 16] for finite access structures, we deduce that for every i the total share of $P_i \triangleq \{p_{a_1}, \ldots, p_{a_i}\}$ is at least the number of sets in the sequence contained in P_i. As in [35], we schedule the parties in B to appear sparsely in $\{p_i\}_{i\in\mathbb{N}}$ and get a lower bound on the total share size of the first t parties in the evolving access structure.

We use the above general lower bound to get lower bounds for two interesting families of access structures. We first construct an infinite independent sequence for an infinite directed st-connectivity access structure. Specifically, we consider a layered graph with 3 layers. Interestingly, we also obtain a lower bound of $\Omega(\sqrt{n})$ on the share size of finite (i.e., not evolving) directed st-connectivity, by taking finite prefixes of the infinite independent sequence. We also prove lower bounds for infinite k-hypergraph access structures for constant k; this is done by generalizing the finite independent sequence for finite k-hypergraph access structures given in [6]. For example, for $k = 3$, we consider the independent sequence that contains all subsets of A of size 2 (hence the set $A_i \cup \{b_i\}$ is of size 3 as required for 3-hypergraph access structures). The number of subsets of A of size 2 contained in $\{p_{a_1}, \ldots, p_{a_i}\}$ is $\Theta(i^2)$; we deduce that the total share size of the first t parties in any evolving secret-sharing scheme realizing this access structure is $\Omega(t^2)$. By a fairly simple construction of an evolving k-hypergraph secret-sharing scheme, our lower bound is tight for k-hypergraph access structures.

Organization. In Section 2, we define secret-sharing schemes and evolving secret-sharing schemes and describe some prior results used in this paper. In Section 3, we define LIBPs and show how to realize them. In Section 4, we construct evolving dynamic threshold secret-sharing schemes with high threshold and in Section 5 we construct evolving secret-sharing schemes for 2-slice and 3-slice access structures. Finally, in Section 6 we prove lower bounds on the size of the shares for st-connectivity and hypergraph access structures.

2 Preliminaries

In this section, we present formal definitions of secret-sharing schemes and evolving secret-sharing schemes.

Notations. For $n \in \mathbb{N}$ we use the notation $[n]$ to denote the set $\{1, 2, \ldots, n\}$. We denote by log the logarithmic function with base 2. When we refer to a set of parties $A = \{p_{i_1}, p_{i_2}, \ldots, p_{i_t}\}$, we assume that $i_1 < i_2 < \cdots < i_t$. We let $\mathrm{poly}(t)$ denote an unspecified polynomial.

2.1 Secret-Sharing Schemes

We start by defining (perfect) secret-sharing schemes for a finite set of parties.

Definition 2.1 (Access Structures)**.** *Let $P = \{p_1, \ldots, p_n\}$ be a set of parties. A collection $\Gamma \subseteq 2^{\{p_1,\ldots,p_n\}}$ is monotone if $B \in \Gamma$ and $B \subseteq C$ imply that $C \in \Gamma$. An access structure $\Gamma \subseteq 2^{\{p_1,\ldots,p_n\}}$ is a monotone collection of non-empty sets. Sets in Γ are called authorized, and sets not in Γ are called unauthorized. We will represent an n-party access structure by a function $f : \{0,1\}^n \to \{0,1\}$, where an input (i.e., a string) $\sigma = \sigma_1, \sigma_2, \ldots, \sigma_n \in \{0,1\}^n$ represents the set $A_\sigma = \{p_i : i \in [n], \sigma_i = 1\}$, and $f(\sigma) = 1$ if and only if $A \in \Gamma$. We will also call f an access structure.*

A secret-sharing scheme defines a way to distribute shares to parties. Such a scheme is said to realize an access structure Γ if the shares held by any authorized set of parties (i.e., a set in the access structure) can be used to reconstruct the secret, and the shares held by any unauthorized set of parties reveal nothing about the secret. The formal definition is given as follows.

Definition 2.2 (Secret-Sharing Schemes)**.** *A secret-sharing scheme Π over a set of parties $P = \{p_1, \ldots, p_n\}$ with domain of secrets S and domain of random strings R is a mapping from $S \times R$ to a set of n-tuples $S_1 \times S_2 \times \cdots \times S_n$ (the set S_j is called the domain of shares of p_j). A dealer distributes a secret $s \in S$ according to Π by first sampling a random string $r \in R$ with uniform distribution, computing a vector of shares $\Pi(s; r) = (\mathsf{sh}_1, \ldots, \mathsf{sh}_n)$, and privately communicating each share sh_j to party p_j. For a set $A \subseteq \{p_1, \ldots, p_n\}$, we denote $\Pi_A(s; r)$ as the restriction of $\Pi(s; r)$ to its A-entries (i.e., the shares of the parties in A).*

A secret-sharing scheme Π with domain of secrets S realizes an access structure Γ if the following two requirements hold:

Correctness. *The secret s can be reconstructed by any authorized set of parties. That is, for any authorized set $B = \{p_{i_1}, \ldots, p_{i_{|B|}}\} \in \Gamma$, there exists a reconstruction function $\mathrm{Recon}_B : S_{i_1} \times \cdots \times S_{i_{|B|}} \to S$ such that for every secret $s \in S$ and every random string $r \in R$, it holds that $\mathrm{Recon}_B(\Pi_B(s; r)) = s$.*

Security. *Every unauthorized set cannot learn anything about the secret from its shares. Formally, for any set $T \notin \Gamma$, every two secrets $s_1, s_2 \in S$, and every possible vector of shares $\langle \mathsf{sh}_j \rangle_{p_j \in T}$,*

$$\Pr\left[\Pi_T(s_1; r) = \langle \mathsf{sh}_j \rangle_{p_j \in T}\right] = \Pr\left[\Pi_T(s_2; r) = \langle \mathsf{sh}_j \rangle_{p_j \in T}\right],$$

where the probability is over the choice of r from R with uniform distribution.

The size of the share of party p_j is defined as $\log |S_j|$ and the size of the shares of Π is defined as $\max_{1 \le j \le n} \log |S_j|$. The total share size of Π is defined as $\sum_{j=1}^{n} \log |S_j|$.

We will use the following result on a construction of secret-sharing schemes from monotone formulas in our constructions.

Theorem 2.3 (Secret Sharing from Monotone Formulas [14])**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a monotone function (i.e., an access structure). If there is a monotone formula with m leaves that computes f, then f can be realized by a secret-sharing scheme in which the total share size is m.*

Below we list the access structures that are of interest in this paper.

Threshold Access Structures. The most basic and well-known access structure is the threshold access structure:

Definition 2.4 (Threshold Access Structures)**.** *Let $1 \leq k \leq n$. A* k-out-of-n threshold access structure Γ *over a set of parties $P = \{p_1, \ldots, p_n\}$ is the access structure containing all subsets of size at least k, that is, $\Gamma = \{A \subseteq P : |A| \geq k\}$.*

The well-known scheme of Shamir for the k-out-of-n threshold access structure (based on polynomial interpolation) is an efficient threshold secret-sharing scheme; the properties of Shamir's scheme over $\mathbb{F}_{2^m}$ for an appropriate $m \in \mathbb{N}$ are summarized in the next theorem.

Theorem 2.5 (Shamir [39])**.** *For every $n \in \mathbb{N}$, and $k \in [n]$, there is a secret-sharing scheme for secrets of size ℓ (i.e., the domain of secrets is $S = \{0,1\}^\ell$) realizing the k-out-of-n threshold access structure, in which the share size is $\max\{\ell, \lceil \log(n+1) \rceil\}$. Moreover, the shares of the scheme are elements of the field $\mathbb{F}_{2^{\ell+\log n}}$.*

st-connectivity Access Structures. The second access structure we consider is the st-connectivity access structure. It is defined as follows.

Definition 2.6 (The Undirected/Directed st-connectivity Access Structures)**.** *Let $G = (V, E)$ be the complete undirected (respectively directed) graph such that $u_s, u_t \in V$. We define the* st-connectivity access structure *as follows: The parties correspond to edges of G. A set of parties is authorized if and only if it contains an undirected (respectively a directed) path from u_s to u_t.*

Benaloh and Rudich [13] constructed a secret-sharing scheme for undirected st-connectivity in which the secret and each share is a bit. The best known secret-sharing scheme for directed st-connectivity is the formula-based scheme of [14] and has share size $n^{O(\log n)}$ (we describe the monotone formula for st-connectivity in Claim 3.16).

Access Structures Defined by Hypergraphs. A *hypergraph* is a pair $H = (V, E)$ where V is a set of vertices and $E \subseteq 2^V \setminus \{\emptyset\}$ is the set of hyperedges. A *hypergraph* where all hyperedges are of size exactly k is called a k-hypergraph. We say H is finite if V is finite. A k-partite hypergraph is a k-hypergraph $H = (V, E)$, for which there is a partition of V to k disjoint sets $D_1, \ldots, D_k$ such that every hyperedge $e \in E$ satisfies $|e \cap D_i| = 1$ for every $1 \leq i \leq k$ (i.e., each hyperedge contains exactly one vertex from each part). An access structure Γ is a k-hypergraph access structure (also called k-homogeneous access structure) if all minimal authorized sets are of size exactly k (described by a hypergraph). Formally, it is defined as follows.

Definition 2.7 (k-Hypergraph Access Structures)**.** *An access structure Γ is a* k-hypergraph access structure *if there exists a finite k-hypergraph $H = (V, E)$ such that the parties of Γ are the vertices V and a set of parties is authorized if and only if it contains a hyperedge (in other words, the minimumal authorized sets of Γ are the hyperedges). An access structure Γ is a k-partite access structure if its k-hypergraph is k-partite.*

Every k-hypergraph access structure with n parties has a monotone formula of size $O(n^k/\log n)$ (by using a result of [26]), thus it can be realized by the secret-sharing scheme of [14] with total share size $O(n^k/\log n)$.

k-Slice Access Structures. An access structure Γ is a k-slice access structure if all sets of size at most $k-1$ are unauthorized, all sets of size at least $k+1$ are authorized, and sets of size k can be either authorized or unauthorized; we describe the authorized sets of size k by a k-hypergraph.

Definition 2.8 (k-Slice Access Structures)**.** *An access structure Γ is a* k-slice access structure *if there exists a finite k-hypergraph $H = (V, E)$ such that the parties of Γ are the vertices V and a set of parties is authorized if and only if it contains at least $k+1$ parties or the sets contains a hyperedge (in other words, the minimal authorized sets of Γ are the hyperedges and all sets of size $k+1$ that do not contain a hyperedge).*

We will use the following constructions for finite slice access structures as a building block in our evolving secret-sharing schemes for infinite slice access structures. They are implied by the CDS protocols of [33, 34] and the transformation of [7, 1].

Theorem 2.9. *Let Γ be a (finite) 2-slice access structure with n parties. Then there is a secret-sharing scheme realizing Γ, in which the share size of every party is at most $2^{O(\sqrt{\log n \cdot \log\log n})}$.*

Theorem 2.10. *Let $k \geq 2$ and let Γ be a (finite) k-slice access structure with n parties. Then there is a secret-sharing scheme realizing Γ, in which the share size of every party is at most $2^{O(\sqrt{\log n} \cdot \log\log n)}$.*

2.2 Evolving Secret-Sharing Schemes

In an evolving secret-sharing scheme, defined by [30], the number of parties is unbounded. Parties arrive one after the other; when a party p_t arrives the dealer gives it a share. The dealer cannot update the share later and does not know how many parties will arrive after party p_t. Thus, we measure the share size of p_t as a function of t. We start by defining an evolving access structure, which specifies the authorized sets. The number of parties in an evolving access structure is infinite, however every authorized set is finite.

Definition 2.11 (Evolving Access Structures)**.** *Let $P = \{p_i\}_{i\in\mathbb{N}}$ be an infinite set of parties. A collection of finite sets $\Gamma \subseteq 2^P$ is an* evolving access structure *if for every $t \in \mathbb{N}$ the collections $\Gamma^t \triangleq \Gamma \cap 2^{\{p_1,\ldots,p_t\}}$ is an access structure as defined in Definition 2.1. We will represent an access structure by a function $f : \{0,1\}^* \to \{0,1\}$, where an input (i.e., a string) $\sigma = \sigma_1, \sigma_2, \ldots, \sigma_n \in \{0,1\}^n$ represents the set $A_\sigma = \{p_i : i \in [n], \sigma_i = 1\}$,[3] and $f(\sigma) = 1$ if and only if $A_\sigma \in \Gamma$. We will also call f an evolving access structure.*

Definition 2.12 (Evolving Secret-Sharing Schemes)**.** *Let S be a domain of secrets, where $|S| \geq 2$, and $\{R_t\}_{t\in\mathbb{N}}, \{S_t\}_{t\in\mathbb{N}}$ be two sequences of finite sets. An* evolving secret-sharing scheme *with domain of secrets S is a sequence of mappings $\Pi = \left\{\Pi^t\right\}_{t\in\mathbb{N}}$, where for every $t \in \mathbb{N}$, Π^t is a mapping $\Pi^t : S \times R_1 \times \cdots \times R_t \to S_t$ (this mapping returns the share sh_t of p_t).*

An evolving secret-sharing scheme $\Pi = \left\{\Pi^t\right\}_{t\in\mathbb{N}}$ realizes an evolving access structure Γ if for every $t \in \mathbb{N}$ the secret-sharing scheme $\Pi_t(s; (r_1, \ldots, r_t)) \triangleq \left\langle \Pi^1(s; r_1), \ldots, \Pi^t(s; r_1, \ldots, r_t) \right\rangle$ (i.e., the shares of the first t parties) is a secret-sharing scheme realizing Γ^t according to Definition 2.2.

[3] In particular, the same set has infinitely many representations by inputs of various lengths, using sufficiently many trailing zeros.

By default, the domain of secrets of an evolving secret-sharing scheme is $\{0,1\}$. The following result shows that every evolving access structure can be realized by an evolving secret-sharing scheme (with exponentially long secrets).

We next define the evolving access structures that we will consider in this paper; they generalize the finite access structures defined in Section 2.1.

Definition 2.13 (Evolving Threshold Access Structures)**.** *Let $k \in \mathbb{N}$. The* evolving k-threshold access structure *is the evolving access structure Γ, where Γ^t is the k-out-of-t threshold access structure.*

Komargodski et al. [30] showed that any evolving threshold access structure can be realized by an efficient evolving secret-sharing scheme.

Theorem 2.14 ([30])**.** *For every $k \in \mathbb{N}$, there is a secret-sharing scheme realizing the evolving k-threshold access structure such that the share size of party p_t is $(k-1) \cdot \log t + \mathrm{poly}(k) \cdot o(\log t)$.*

A natural generalization of an evolving threshold access structure is to allow the threshold to depend on the index of the arriving party.

Definition 2.15 (dynamic-threshold Access Structures)**.** *Let $\mathrm{tr} : \mathbb{N} \to \mathbb{N}$ be a non-decreasing function. A* $\mathrm{tr}(t)$-dynamic-threshold access structure *is defined as follows: A finite set of parties A is authorized if and only if there exists t such that $|A \cap \{p_1, \ldots, p_t\}| \geq \mathrm{tr}(t)$. Stated differently, a set is unauthorized A if and only if for every t, it holds that $|A \cap \{p_1, \ldots, p_t\}| < \mathrm{tr}(t)$.*

Komargodski and Paskin-Cherniavsky [31] showed that any dynamic-threshold access can be realized with an evolving secret-sharing scheme with a polynomial share size.

Theorem 2.16 ([31])**.** *For any dynamic-threshold access structure, there exists an evolving secret-sharing scheme in which the share size of party p_t is $O(t^4 \cdot \log t)$.*

Definition 2.17 (Evolving Undirected/Directed st-connectivity Access Structures)**.** *An evolving undirected (resp. directed) st-connectivity access structure is defined as follows. The parties in the access structure are the edges of an undirected (resp. directed) graph $G = (V, E)$, where V is countably infinite, with some order on the edges that specifies the order that the parties arrive. There are two fixed vertices in the graph $u_s, u_t \in V$, where u_s is called the source vertex and u_t the target vertex. A finite set of parties (i.e., edges) is authorized if and only if they contain an undirected (resp. a directed) path from u_s to u_t.*

Komargodski et al. [30] showed that every undirected st-connectivity access structure can be realized by an evolving secret-sharing scheme in which the share of each party is a bit.

Evolving k-Hypergraph access structures and *evolving k-slice access structures* are defined as their finite counterparts, except that the k-hypergraph H is countably infinite. In these access structures, we assume that there is some order on the vertices of the hypergraph that specifies the order that the parties (i.e., vertices) arrive.

3 Evolving Secret-Sharing Schemes for Infinite Branching Programs

Infinite decision trees were used in [30, 31] to construct evolving secret-sharing schemes. In this section, we define infinite non-deterministic branching programs (see Section 3.1.1), show how to

transform them to generalized infinite decision trees (see Section 3.1.3), and show how to construct secret-sharing schemes for infinite decision trees (see Section 3.1.2). This setting will be used in our constructions of schemes for various functions, i.e., for evolving access structures that have infinite branching programs with bounded-width (see Section 3.3).

3.1 Constructing an Evolving Secret-Sharing Schemes for Infinite Branching Programs

3.1.1 Infinite Branching Programs and Generalized Infinite Decision Trees

An infinite monotone non-deterministic branching program (IBP) is a generalization of finite monotone non-deterministic branching programs except that the number of edges and variables is infinite. Such a branching program computes a monotone function $f : \{0,1\}^* \to \{0,1\}$ on all finite inputs; i.e., defines an evolving access structure (see Definition 2.11).

Recall that a finite monotone non-deterministic branching program is a directed acyclic graph, where each edges is labeled by a variable from $x_1, \ldots, x_t$. For every input $\sigma \in \{0,1\}^t$ (interpreted as an assignment to the variables $x_1, \ldots, x_t$) it holds that $f(\sigma) = 1$ if and only if *there exists* a path in G from the source vertex to a target vertex that is satisfied by σ, that is, each edge in the path is labeled with a variable x_i that is assigned 1, i.e., $\sigma_i = 1$. Below we generalize this notion to infinite branching programs. In this definition we will allow many target vertices and we will allow the edges to be labeled by 1, meaning that every assignment satisfies them.

Definition 3.1 (Infinite Monotone Non-Deterministic Branching Programs – IBP)**.** *An infinite monotone non-deterministic branching program is a triple $B = (G = (V, E), u_0, \mu)$, where V is a countable set of vertices, G is an infinite directed acyclic graph, u_0 is a source vertex, and $\mu : E \to \{x_i : i \in \mathbb{N}\} \cup \{1\}$ is a labeling of the edges by variables or by 1 (we will sometimes use the notation μ_e instead of $\mu(e)$). We denote by U_{leaf} the set of vertices in V with out-degree 0, i.e., the leaves.*

For a path P in the branching program and an input (an assignment for the variables on the path) $\sigma \in \{0,1\}^t$ for some $t \in \mathbb{N}$, we say that σ satisfies P and denote $\text{sat}_P(\sigma) = 1$ if σ satisfies all variables on the path, i.e., for each edge e on the path either $\mu_e = 1$ or $\mu_e = x_i$ for some $1 \leq i \leq t$ such that $\sigma_i = 1$. The branching program accepts an input σ if there exists a directed path P starting in the source vertex u_0 and leading to some leaf $u \in U_{\text{leaf}}$ such that $\text{sat}_P(\sigma) = 1$. The function $f : \{0,1\}^* \to \{0,1\}$ computed by B is the function f such that $f(\sigma) = 1$ if and only if B accepts σ.

To clarify what it means for an assignment σ to satisfy a path P, consider as an example $P = (e_1, e_2, e_3, e_4)$, with respective labels $(x_7, x_3, 1, x_{25})$. Then, the assignment $\alpha \in \{0,1\}^{30}$ with $\alpha_i = 1$ if and only if i is odd satisfies P, an assignment $\beta \in \{0,1\}^{25}$ with $\beta_3 = 0$ does not satisfy P, and the assignment $\gamma = 1^{20}$ also does not satisfy P, since it is too short.

We will construct evolving secret-sharing schemes for IBPs that are layered as defined below. Intuitively, a layered IBP is a restriction of IBP, defined over layered (infinite, directed, acyclic) graphs (i.e., with edges going only from any one layer to its consecutive layer) with the additional requirement that the label of any edge from layer $i-1$ to layer i is either x_i or 1.

Definition 3.2 (Layered IBP)**.** *An infinite monotone non-deterministic branching program (abbreviated LIBP) is* layered *if the vertices of G can be partitioned into finite sets $(L_i)_{i \in \mathbb{N}}$, such that $L_0 = \{u_0\}$, there are edges only from layer $i-1$ to layer i for some $i \in \mathbb{N}$, and all edges entering*

layer i are labeled either by x_i or by 1. For a vertex $u \in V$, we denote $L(u)$ as the layer of u, i.e., the index i such that $u \in L_i$. The width of the branching program at layer i, denoted $w(i)$, is the number of vertices in layer L_i. For a LIBP $B = (G, u_0, \mu)$ and vertices u, v, we define the predicate $\mathrm{reach}_{u,v}$ *as* $\mathrm{reach}_{u,v} = 0$ *if there is no path in G from u to v, and otherwise*

$$\mathrm{reach}_{u,v}(x_{L(u)+1}, \ldots, x_{L(v)}) = \bigvee_{\substack{P \text{ is a path in } G \\ \text{from } u \text{ to } v}} \mathrm{sat}_P(x_{L(u)+1}, \ldots, x_{L(v)}),$$

i.e., an input satisfies $\mathrm{reach}_{u,v}$ *if and only if it satisfies at least one path from u to v. We stress that, unlike Definition 3.1, here we consider an assignment to the predicate* sat_P *that only contains the variables that can appear on the path, i.e., we consider assignments $\sigma_{L(u)+1}, \ldots, \sigma_{L(v)}$ to the variables $x_{L(u)+1}, \ldots, x_{L(v)}$.*

Our goal is to construct an evolving secret-sharing scheme for the access structure described by a LIBP B; we call such a scheme an evolving secret-sharing realizing B.

Example 3.3. We next describe an LIBP $B = (G = (V, E), u_0^0, \mu)$ for the 3-threshold function access structure (as defined in Definition 2.15). The layers of the graph are $L_t = \{u_0^t, \ldots, u_{\min\{3,t\}}^t\}$ for $t \in \mathbb{N} \cup \{0\}$ and the vertices of the graph are $V = \cup_{t \in \mathbb{N} \cup \{0\}} L_t$. For every $t \in \mathbb{N}$ and $0 \leq i \leq 2$ there are two edges: an edge (u_i^{t-1}, u_i^t) labeled by 1 and an edge (u_i^{t-1}, u_{i+1}^t) labeled by x_t. Notice that the leaves are $U_{\text{leaf}} = \{u_3^t : t \in \mathbb{N}, t \geq 3\}$. An illustration of this construction appears in Figure 1.

Informally, reaching the vertex u_i^t in the LIBP means that an input σ of length t contains at least i ones. Indeed, let $\sigma \in \{0,1\}^t$ be an input that contains $i < 3$ ones. Then, any path in G that is satisfied by σ contains at most i edges that are not 1-labeled, and hence ends in a node of the form $u_j^{t'}$ for $j < 3$, i.e., not in any leaf. Conversely, let $\sigma \in \{0,1\}^t$ be an input that contains $i \geq 3$ ones, and let $t_1 < t_2 < t_3$ be the first three coordinates in σ that are 1. It is possible to to define a path from u_0^0 to the leaf $u_3^{t_3}$ that is satisfied by σ, as follows. At each step the 1-labeled edge is taken, except for steps t_1, t_2, and t_3, at which the edges $(u_0^{t_1-1}, u_1^{t_1})$, $(u_1^{t_2-1}, u_2^{t_2})$, and $(u_2^{t_3-1}, u_3^{t_3}))$ are taken, respectively.

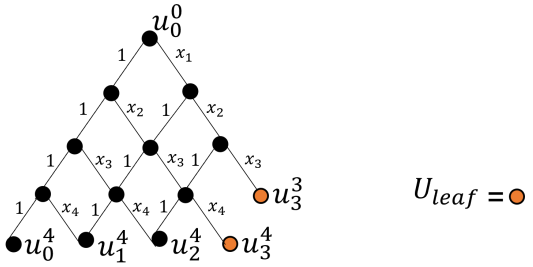

Figure 1: The first five layers of the LIBP for the 3-threshold function. For example, u_3^4 is a leaf and the input $\sigma = 1, 1, 0, 1$ satisfies the path $u_0^0, u_1^1, u_2^2, u_2^3, u_3^4$.

In our constructions, we use *generalized* infinite decision trees. On one hand, we limit the graph to be an infinite tree. On the other hand, each edge, instead of being labeled by a variable x_i, is labeled by a predicate of some variables $x_i, \ldots, x_j$. Being a bit more specific, we divide the variables into generations $\{G_i\}_{i \in \mathbb{N}}$, and let an edge of distance i from the root be labeled with a predicate on the variables in generation G_i.

Definition 3.4 (Generalized Infinite Decision Trees – GIDT)**.** *A generalized infinite decision tree is a quadruple $T = (G = (V, E), u_0, \mu, h)$, where*

- *V is a countable set of vertices,*
- *$G = (V, E)$ is an infinite directed tree with root vertex u_0 such that the out-degree of each vertex is finite. We denote that i^{th} level L_i as $\{u \in V : u$ is at distance i from $u_0\}$, and refer to L_i as the i^{th}* layer*.*
- *$h : \mathbb{N} \to \mathbb{N}$ is an increasing function that partitions the variables into generations, where for $i \in \mathbb{N}$, generation i is the variables $G_i \triangleq \{x_{h(i-1)+1}, \ldots, x_{h(i)}\}$ (where we define $h(0) = 0$),*
- *μ is a labeling of the edges by* predicates*, where for every edge e from level L_{i-1} to level L_i, the labeling μ_e is some monotone predicate on the variables of generation i, of the form $\varphi(x_{h(i-1)+1}, \ldots, x_{h(i)}) : \{0,1\}^{h(i)-h(i-1)} \to \{0,1\}$.*

For a path P in the tree ending at a vertex in layer i, we say that P is satisfied by an input $\sigma \in \{0,1\}^t$, denoted by $\text{sat}_P(\sigma) = 1$, if $h(i) \leq t$ (that is, the variables in all predicates labeling edges in P are from $x_1, \ldots, x_t$) and for each edge e on the path the predicate μ_e is satisfied by σ. The GIDT T accepts an input σ if there is at least one directed path P starting in the source vertex u_0 and leading to a leaf such that $\text{sat}_P(\sigma) = 1$. The function $f : \{0,1\}^* \to \{0,1\}$ computed by T is the function f such that $f(\sigma) = 1$ if and only if T accepts σ.

Example 3.5. We show an example of a GIDT $T = (G = (V, E), u_0, \mu, h)$ for a dynamic 3-threshold function (i.e., defined by a function $\text{tr}(t) = 3$). Let $h : \mathbb{N} \to \mathbb{N}$ be any increasing function. The layers of G are

$$L_i = \{u_{b_0,b_1,\ldots,b_i} : 0 = b_0 \leq b_1 \leq \cdots \leq b_{i-1} \leq b_i \leq 3, b_{i-1} \leq 2\}.$$

The vertices of G are $V = \cup_{i \in \mathbb{N} \cup \{0\}} L_i$. There is an edges $(u_{b_0,\ldots,b_{i-1}}, u_{b_0,\ldots,b_{i-1},b_i})$ in G for every $i \in \mathbb{N}$ and every sequence $b_0, b_1, \ldots, b_i$, where $0 = b_0 \leq b_1 \leq \cdots \leq b_{i-1} \leq b_i \leq 3, b_{i-1} \leq 2$, and this edge is labeled by $\text{Thr}_{b_i - b_{i-1}}(x_{h(i-1)+1}, \ldots, x_{h(i)})$. Thr_b is the b-threshold predicate, i.e., $\text{Thr}_b(y_1, \ldots, y_m) = 1$ iff $\sum_{j=1}^m y_j \geq b$. For example, if $b_i = b_{i-1}$, then the label of the edge is 1. The leaves of the GIDT T are $U_{\text{leaf}} = \{u_{b_0,b_1,\ldots,b_i} : i \in \mathbb{N}, 0 = b_0 \leq b_1 \leq \cdots \leq b_{i-1} < b_i = 3\}$. The GIDT is described in Figure 2.

Informally, the GIDT counts the number of ones in an input σ in each generation. More formally, there is a path from the root u_0 to a vertex $u_{b_0,b_1,\ldots,b_i}$ (for $i \in \mathbb{N}$) that is satisfied by an input $\sigma \in \{0,1\}^t$ if and only if σ contains at least $b_j - b_{j-1}$ ones from generation j for each $1 \leq j \leq i$. In particular, T accepts an input σ if and only if there is a path from u_0 to a leaf $u_{b_0,b_1,\ldots,b_i}$, where $b_i = 3$, that is satisfied by σ if and only if σ contains at least 3 ones.

3.1.2 An Evolving Secret-Sharing Scheme for GIDTs

We next present an evolving secret-sharing scheme for GIDTs. As a first step, we present an evolving secret-sharing scheme for simple infinite decision trees, defined next.

Definition 3.6 (Infinite decision trees – IDT)**.** *An infinite decision tree $T = (G = (V, E), u_0 = 0, \mu)$ is a special case of GIDT, where each edge (u, v) is either labeled by the constant 1 or by a variable x_v, where for simplicity we assume that $V = \mathbb{N} \cup \{0\}$ (i.e., a vertex is a non-negative integer). As G is a tree, each variable labels at most one edge. Furthermore, we assume that the vertices are*

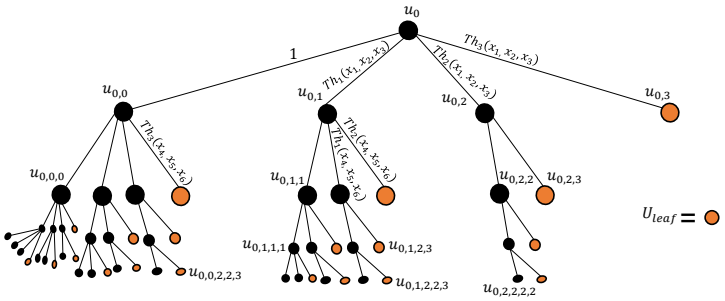

Figure 2: The first five layers of the GIDT for 3-threshold function where $h(0) = 0$ and $h(t) = 3t$.

ordered by the layers, i.e., $L_0 = \{0\}$, $L_1 = \{1, \ldots, w(1)\}$, and so on (where $w(i)$ is the width of layer L_i). The variables in generation i are $\{x_j : j \in L_i\}$ (thus, we do not need to specify h for an IDT).

We note that IDT is not a special case of LIBP, since in IDT different edges of the same layer may be labeled by different variables, where in LIBP labels from layer i are labeled either by 1 or by x_{i+1}.

We next recall the evolving secret-sharing scheme for infinite decision trees from [30, 31]; the scheme we present also deals with edges labeled by 1. We will use this scheme to construct an evolving secret-sharing scheme for GIDTs and LIBPs. For technical reasons we assume that all edges entering leaves are labeled by a variable.[4]

Construction 3.7 (An Evolving Secret-Sharing Scheme Π_{IDT} for an IDT $T = (G, u_0, \mu)$)**.**
Input: $s \in \{0, 1\}$.
The sharing algorithm:

- For $i = 1$ to ∞:
  - For every vertex $u \in L_{i-1}$ and $v \in L_i$, when party p_v arrives choose a bit r_v as follows:
    - If v is a leaf, then let $u_0, v_1, \ldots, v_{t-1}, v$ be the path from the root u_0 to v in G and assign $r_v \leftarrow s \oplus \bigoplus_{j=1}^{t-1} r_{v_j}$.
    - If v is not a leaf and $\mu_{(u,v)} = x_v$, then r_v is a uniformly distributed random bit.
    - If v is not a leaf and $\mu_{(u,v)} = 1$, then $r_v \leftarrow 0$.
  - The share of p_v is $\mathsf{sh}_v = r_v$.

Claim 3.8. *The evolving secret-sharing scheme Π_{IDT} realizes the infinite decision tree $T = (G, u_0, \mu)$, where the share of p_t is a bit.*

Proof. First we prove the correctness of the scheme. Let $\sigma = \sigma_1, \ldots, \sigma_t$ be an input accepted by T, where $\sigma_1, \ldots, \sigma_{t-1}$ is not accepted by T (in particular, $\sigma_t = 1$), and let $A = \{p_i : \sigma_i = 1\}$ be the corresponding set of parties. There is a path $u_0, v_1, \ldots, v_{t-1}, t$ in G from u_0 to the leaf t such that

[4] As the function computed by an IDT describes an access structure, we assume that the empty set is rejected by the IDT. For every vertex u in the tree whose in-coming edge is labeled by a variable, if there exists a path from v to a leaf such that all labels on the path are 1, we remove the subtree of v (i.e., v becomes a leaf).

σ satisfies all labels on this path. By our assumption, the edge (v_{t-1}, t) is labeled by x_t. As $\sigma_t = 1$, the party p_t is in A and the parties in A have the bit $r_t = s \oplus \bigoplus_{j=1}^{t-1} r_{v_j}$. Furthermore, for every j either $\mu_{(v_{j-1},v_j)} = 1$ and $r_{v_j} = 1$ or $\mu_{(v_{j-1},v_j)} = x_{v_j}$, thus, $\sigma_j = 1$ and the parties in A hold r_{v_j}. We conclude that the parties in A can reconstruct s.

Next we prove the security of the scheme. It would be convenient to view the scheme Π_{IDT} as a recursive procedure. To share a secret s in Π_{IDT} for the subtree of T rooted at a vertex u, the dealer independently executes a secret-sharing scheme for each vertex v such that $(u, v) \in E$:

- If $\mu_{(u,v)} = 1$, the dealer shares s recursively for the subtree of G rooted at v.
- If $\mu_{(u,v)} = x_v$, the dealer chooses a random bit r_v, gives r_v to p_v, and shares $s \oplus r_v$ recursively for the subtree of G rooted at v.

(If u is a leaf then there are no recursive calls.) Let $\sigma = \sigma_{u+1}, \ldots, \sigma_t$ be an input not accepted by the subtree of T rooted at u, and let $A = \{p_i : \sigma_i = 1\}$ be the corresponding set of parties. We prove the security, that is, that the parties in A learn no information on the secret, by induction on $|\sigma|$. For the basis case when $|\sigma| = 1$, the vertex t is not a leaf and the share of p_t is either 1 or a random bit. For the induction step, as the infinite decision tree rooted at u rejects the input $\sigma = \sigma_u, \ldots, \sigma_t$, there is no path from u to a leaf that is satisfied by σ. We will show that the set A does not learn information from the secret-sharing scheme for each vertex v such that $(u, v) \in E$. Since each secret-sharing scheme is independently executed, this will imply the security. Fix such a vertex v. If $\mu_{(u,v)} = 1$, there is no path from v to a leaf that is satisfied by $\sigma_{v+1}, \ldots, \sigma_t$; by induction, the set $\{p_i : v + 1 \leq i \leq t, \sigma_i = 1\}$ learns no information on s from this execution (and the shares of $A \setminus \{p_i : v + 1 \leq i \leq t, \sigma_i = 1\}$ are independent of the shares of the recursive call to v). If $\mu_{(u,v)} = x_v$, there are two cases:

- If $\sigma_v = 1$, then there is no path from v to a leaf that is satisfied by $\sigma_{v+1}, \ldots, \sigma_t$; by induction, the set $A \setminus \{p_v\}$ learns no information on $s \oplus r_v$ from this execution, hence learns no information on s.
- If $\sigma_v = 0$, then $p_v \notin A$, and the set A learns no information on r_v, thus although it might learn $s \oplus r_v$, it learns no information on s.

□

We next show how to realize the access structure of a GIDT using the secret-sharing scheme Π_{IDT} realizing a related infinite decision tree (where edges are labeled by variables or by the constant 1). In a GIDT each edge e is labeled by a predicate a_e; in the following scheme Π_{GIDT} we consider this predicate as describing an access structure over the parties of the generation.

Construction 3.9 (An Evolving Secret-Sharing Scheme Π_{GIDT} for a GIDT $T = (G = (V, E), u_0, \mu, h)$)**.**

Input: $s \in \{0, 1\}$.

- Construct from the GIDT $T = (G = (V, E), u_0, \mu, h)$ an IDT $T' = (G = (V, E), u_0, \mu')$ whose variables are $\{y_i : i \in \mathbb{N}\}$, where for every edge $(u, v) \in E$ if the predicate $\mu_{(u,v)}$ is the constant predicate 1, then $\mu'(u, v) = 1$; otherwise $\mu'(u, v) = y_v$.
- Execute the scheme Π_{IDT} for T' and use its shares as follows:

(∗ Recall that in Π_{IDT} the the parties arrive according to layers, where inside a layer the order is some arbitrary fixed order ∗)

- For $i = 1$ to ∞ do:
  - When party $p_{h(i-1)+1}$ arrives do:
    - For every $(u, v) \in E$, where $u \in L_{i-1}$, $v \in L_i$, and $\mu_{(u,v)} \neq 1$, generate the share r_v of y_v in the scheme Π_{IDT} and share r_v using a secret-sharing scheme realizing the access structure defined by $\mu_{(u,v)}$ among the parties $p_{h(i-1)+1}, \ldots, p_{h(i)}$.
    - Let sh_t, for $h(i-1)+1 \leq t \leq h(i)$, be the concatenation of the shares of p_t in all these schemes.
    - Give party $p_{h(i-1)+1}$ the share $\mathsf{sh}_{h(i-1)+1}$.
  - For $t = h(i-1)+2$ to $h(i)$ do:
    - When party p_t arrives give it the share sh_t.

Claim 3.10. *The evolving secret-sharing scheme Π_{GIDT} realizes the GIDT $T = (G, u_0, \mu, h)$. For a party p_t in generation i (that is, $h(i-1)+1 \leq t \leq h(i)$), the size of the share of p_t is the sum of the sizes of the shares of p_t in the secret-sharing schemes for $\mu_{(u,v)}$ for every $(u, v) \in E$ such that $u \in L_{i-1}$, $v \in L_i$, and $\mu_{(u,v)} \neq 1$ (there are at most $w(i)$ such schemes).*

Proof. First we prove the correctness of the scheme. Let σ be an input accepted by T and $A = \{p_i : \sigma_i = 1\}$. Thus, there exists an accepting a path P from u_0 to a leaf such that $\mathrm{sat}_P(\sigma) = 1$. For every edge $e = (u, v) \in P$, the input σ satisfies μ_e. If $\mu_e \neq 1$, the parties in A can reconstruct r_v using the shares of the secret-sharing scheme realizing μ_e. If $\mu_e = 1$, according Construction 3.7, $r_v = 0$. By the correctness of Construction 3.7, the parties in A can reconstruct s by computing the exclusive-or of the bits of the vertices on P.

Next we prove the security of the scheme. Let σ be an input rejected by T and $A = \{p_i : \sigma_i = 1\}$. The parties in A can reconstruct the shares r_v in Π_{IDT} for edges (u, v) such that $\mu_{(u,v)}(\sigma) = 1$ and do not get any information on shares r_v for edge (u, v) such that $\mu_{(u,v)}(\sigma) = 0$. Since σ is rejected by T, there does not exists an accepting a path P from u_0 to a leaf such that $\mathrm{sat}_P(\sigma) = 1$. That is, the parties in A hold shares in Π_{IDT} of an unauthorized set in T'. By the security of Π_{IDT}, the shares r_v that the parties in T can reconstruct are equally distributed when $s = 0$ and when $s = 1$.

For the share size, party p_t obtains a share in the secret-sharing scheme realizing $\mu_{(u,v)}$ for each edge (u, v) where $u \in L_{i-1}$ and $v \in L_i$. □

3.1.3 A Transformation from LIBPs to GIDTs

We next describe a transformation from an LIBP B to a GIDT T computing the same function. We start with an informal description of the transformation. To transform an LIBP B to an IDT T (where each edge is labeled by a variable or the constant 1), we duplicate vertices and have in T a vertex $u_{0,j_1,\ldots,j_{i-1},j_i}$ for every path $u_0, u_{j_1}, \ldots, u_{j_{i-1}}, u_{j_i}$ in T starting from the root, and add an edge $(u_{0,j_1,\ldots,j_{i-1}}, u_{0,j_1,\ldots,j_{i-1},j_i})$ whose label is the label of the edge $(u_{j_{i-1}}, u_{j_i})$. The problem with this construction is that the resulting GIDT is to big. To construct more efficient GIDT (which will result in more efficient evolving secret-sharing schemes), we partition the variables into generations (described by a function $h : \mathbb{N} \to \mathbb{N}$), the vertices in layer i of T are $u_{0,j_1,j_2,\ldots,j_i}$ for vertices $u_0, u_{j_1}, u_{j_2}, \ldots, u_{j_i}$ in the layers $0, h(1), h(2), \ldots, h(i)$ in B respectively. That is, the

number of vertices in the resulting GIDT is much smaller. Now an edge $(u_{0,j_1,\ldots,j_{i-1}}, u_{0,j_1,\ldots,j_{i-1},j_i})$ representing *all paths* in B from $u_{j_{i-1}}$ to u_{j_i}, i.e., the predicate of this edge is satisfied by an input σ if and only if σ satisfies some path in B from $u_{j_{i-1}}$ to u_{j_i}. The formal construction is described below.

Construction 3.11 (A Transformation from a LIBP to a GIDT)**.**
Input: A LIBP $B = (G = (V, E), u_0, \mu)$ and an increasing function $h : \mathbb{N} \to \mathbb{N}$. ($*$ We use the following notation for the vertices of the LIBP B – the vertices in the i-layer of B are $L_i = \{u^i_1, \ldots, u^i_{w(i)}\}$ for $i \in \mathbb{N} \cup \{0\}$. $*$)
Output: A GIDT $T = (G' = (V', E'), u'_0, \mu', h)$.
The transformation:

- The vertices in layer i of the tree G' are $L'_0 = \{u_0\}$ and for $i \in \mathbb{N}$ define

$$L'_i = \{u_{0,j_1,\ldots,j_i} : 1 \le j_1 \le w(h(1)), j_1 \notin U_{\text{leaf}}, \ldots, 1 \le j_i \le w(h(i)), j_i \notin U_{\text{leaf}}\} \bigcup \left\{v_{0,j_1,\ldots,j_{i-1}} : 1 \le j_1 \le w(h(1)), j_1 \notin U_{\text{leaf}}, \ldots, 1 \le j_{i-1} \le w(h(i-1)), j_{i-1} \notin U_{\text{leaf}}\right\}.$$

  The vertices of G are $V' = \cup_{i \in \mathbb{N} \cup \{0\}} L'_i$. The leaves are $U'_{\text{leaf}} = \cup_{i \in \mathbb{N}} \{v_{0,j_1,\ldots,j_{i-1}} : 1 \le j_1 \le w(h(1)), j_1 \notin U_{\text{leaf}}, \ldots, 1 \le j_{i-1} \le w(h(i-1)), j_{i-1} \notin U_{\text{leaf}}\}$.

- The edges are

$$E' = \left\{(u_{0,j_1,\ldots,j_{i-1}}, u_{0,j_1,\ldots,j_{i-1},j_i}) : i \in \mathbb{N}, u_{0,j_1,\ldots,j_{i-1},j_i} \in V'\right\} \bigcup \left\{(u_{0,j_1,\ldots,j_{i-1}}, v_{0,j_1,\ldots,j_{i-1}}) : i \in \mathbb{N}, u_{0,j_1,\ldots,j_{i-1}} \in V'\right\}.$$

- For every $e = \left(u_{0,j_1,\ldots,j_{i-1}}, u_{0,j_1,\ldots,j_{i-1},j_i}\right) \in E'$, let $u = u^{h(i-1)}_{j_{i-1}}$, $v = u^{h(i)}_{j_i}$ and define

$$\mu'_e(x_{h(i-1)+1}, \ldots, x_{h(i)}) = \text{reach}_{u,v}(x_{h(i-1)+1}, \ldots, x_{h(i)}).$$

- For every $e = \left(u_{0,j_1,\ldots,j_{i-1}}, v_{0,j_1,\ldots,j_{i-1}}\right) \in E'$, let $u = u^{h(i-1)}_{j_{i-1}}$ and define

$$\mu'_e(x_{h(i-1)+1}, \ldots, x_{h(i)}) = \bigvee_{\substack{v \text{ is a leaf in layers} \\ h(i-1)+1,\ldots,h(i) \text{ in } B}} \text{reach}_{u,v}(x_{h(i-1)+1}, \ldots, x_{L(v)}).$$

Claim 3.12. *Construction 3.11 outputs a GIDT T which computes the same function as B. Furthermore, the number of vertices in layer i of T is $|L'_i| = \left(\prod_{1 \le j < i}(w(h(j))\right) \cdot (w(h(i)) + 1)$.*

Proof. We first prove the equivalence of B and T, that is, we prove that B accepts an input $\sigma = \sigma_1, \ldots, \sigma_t$ if and only if T accepts σ. Let ℓ be the generation of t, that is $h(\ell-1)+1 \le t \le h(\ell)$.

First assume that B accepts σ. W.l.o.g., assume that no proper prefix of σ is accepted by B (other apply the following argumenets to such minimal prefix). Then, there exists a path $P = (u^0_0, u^1_{j_1}, \ldots, u^t_{j_t})$ in G where $u^t_{j_t} \in U_{\text{leaf}}$ and $\text{sat}_P(\sigma) = 1$. Consider the path

$$P' = (u_0, u_{0,j_{h(1)}}, \ldots, u_{0,j_{h(1)},j_{h(2)},\ldots,j_{h(\ell-1)}}, v_{0,j_{h(1)},j_{h(2)},\ldots,j_{h(\ell-1)}})$$

in G'. We partition the path P in G to sub-paths – for every $1 \leq i \leq \ell-1$, let $P^i = (u^{h(i-1)}_{j_{h(i-1)}}, \ldots, u^{h(i)}_{j_{h(i)}})$ and let $P^\ell = (u^{h(\ell-1)}_{j_{h(\ell-1)}}, \ldots, u^t_{j_t})$. Since $\mathrm{sat}_P(\sigma) = 1$, we deduce that $\mathrm{sat}_{P^i}(\sigma_{h(i-1)+1}, \ldots, \sigma_{h(i)}) = 1$ for $1 \leq i \leq \ell - 1$ and $\mathrm{sat}_{P^\ell}(\sigma_{h(i-1)+1}, \ldots, \sigma_t) = 1$. By the definition of $\mathrm{reach}_{u,v}$, this implies that

$$\mathrm{reach}_{u^{h(i-1)}_{j_{h(i-1)}}, u^{h(i)}_{j_{h(i)}}}(\sigma_{h(i-1)+1}, \ldots, \sigma_{h(i)}) = 1$$

for every $1 \leq i \leq \ell - 1$ and $\mathrm{reach}_{u^{h(\ell-1)}_{j_{h(\ell-1)}}, u^t_{j_t}}(\sigma_{h(i-1)+1}, \ldots, \sigma_t) = 1$, where $u^t_{j_t}$ is a leaf in B. Thus, in T we have

$$\mathrm{sat}_{P'}(\sigma) = \left(\bigvee_{\substack{v \text{ is a leaf in layers} \\ h(\ell-1)+1,\ldots,h(\ell) \text{ in } B}} \mathrm{reach}_{u^{h(\ell-1)}_{j_{h(\ell-1)}}, v}(\sigma_{h(\ell-1)+1}, \ldots, \sigma_{L(v)}) \right)$$
$$\wedge \left(\bigwedge_{1 \leq i \leq \ell-1} \mathrm{reach}_{u^{h(i-1)}_{j_{h(i-1)}}, u^{h(i)}_{j_{h(i)}}}(\sigma_{h(i-1)+1}, \ldots, \sigma_{h(i)}) \right) = 1.$$

In the other direction, assume that T accepts σ. W.l.o.g., assume that no proper prefix of σ is accepted by T. Then in T there exists a path

$$P' = (u_0, u_{0,j_1}, \ldots, u_{0,j_1,\ldots,j_{\ell-1}}, v_{0,j_1,\ldots,j_{\ell-1}})$$

where $v_{0,j_1,\ldots,j_{\ell-2}}$ is a leaf and $\mathrm{sat}_{P'}(\sigma) = 1$. This implies that for every $1 \leq i \leq \ell - 1$

$$\mu'_{(u_{0,j_1,\ldots,j_{i-1}}, u_{0,j_1,\ldots,j_{i-1},j_i})}(\sigma_{h(i-1)+1}, \ldots, \sigma_{h(i)})$$
$$= \mathrm{reach}_{u^{i-1}_{j_{i-1}}, u^i_{j_i}}(\sigma_{h(i-1)+1}, \ldots, \sigma_{h(i)}) = 1.$$

Thus, for each $1 \leq i \leq \ell-1$, there exists some path P^i from $u^{i-1}_{j_i}$ to $u^i_{j_i}$ in G such that $\mathrm{sat}_{P^i}(\sigma) = 1$. Furthermore, since $\mu'_{(u_{0,j_1,\ldots,j_{\ell-1}}, v_{0,j_1,\ldots,j_{\ell-1}})}(\sigma_{h(\ell-1)+1}, \ldots, \sigma_t) = 1$ there exists a leaf v in B such that $\mathrm{reach}_{u^{\ell-1}_{j_{\ell-1}}, v}(\sigma_{h(\ell-1)+1}, \ldots, \sigma_t) = 1$ and, therefore, in G there exists a path P^ℓ from $u^{\ell-1}_{j_{\ell-1}}$ to a leaf v such $\mathrm{sat}_{P^\ell}(\sigma) = 1$. By concatenating the paths $P^1, \ldots, P^\ell$, we obtain a path P in G from u_0 to a leaf for which $\mathrm{sat}_P(\sigma) = 1$; thus, B accepts σ.

To bound $|L'_i|$, recall that a vertex in layer i of T is either $u_{0,j_1,\ldots,j_i}$ or a leaf $v_{0,j_1,\ldots,j_{i-1}}$, thus

$$|L'_i| \leq \left(\prod_{1 \leq j \leq i-1} w(h(i)) \right) \big(w(h(i)) + 1\big).$$

□

3.1.4 Putting Everything Together

Next, we combine our results from Sections 3.1.1 to 3.1.3 and construct an evolving secret-sharing scheme for LIBPs.

Theorem 3.13. *Let $B = (G = (V, E), u_0, \mu)$ be an LIBP and $h : \mathbb{N} \to \mathbb{N}$ be an increasing function, where $h(0) = 0$. There exists an evolving secret-sharing scheme realizing B in which the share of a party p_t in generation i, i.e., $h(i-1)+1 \leq t \leq h(i)$, is the shares of p_t in the $\left(\prod_{1 \leq j \leq i-1} w(h(j))\right)(w(h(i))+1)$ secret-sharing schemes realizing the predicates of the edges between layer $i-1$ and layer i in the GIDT constructed in Construction 3.11.*

Proof. We apply Construction 3.11 to transform B into an equivalent GIDT T with the specified h. Then apply Construction 3.9 to T to obtain an evolving secret-sharing scheme realizing T. By Claim 3.12 and Claim 3.10, we obtain an evolving secret-sharing scheme realizing B with the shares as stated in the theorem. □

Remark 3.14. Theorem 3.13 does not state how to choose the function h for a given LIBP B. The choice of h that will minimize the share size depends on the particular LIBP B. On one hand, when h grows slowly, the number of vertices in each level of the GIDT T obtained from B becomes larger. On the other hand, if h grows fast, there are more parties in each generation and the predicates labeling the edges of T become more complicated. The optimal choice of h should balance these two conflicting complexities.

3.2 Evolving Secret-Sharing Schemes for Dynamic-Threshold via LIBPs

Komargodski and Paskin-Cherniavsky [31] have constructed an evolving secret-sharing scheme for dynamic-threshold access structures; their construction uses GIDTs. As an example of our construction of an evolving secret-sharing schemes for LIBPs, we describe their construction using our framework.

Fix a non-decreasing function $\mathrm{tr} : \mathbb{N} \to \mathbb{N}$. We first describe an LIBP $B_{\mathrm{DynTr}} = (G_{\mathrm{DynTr}} = (V, E), u_0^0, \mu)$ for the $\mathrm{tr}(t)$-dynamic-threshold function access structure (as defined in Definition 2.15); this construction generalized the ideas of Example 3.3. The construction we describe can be optimized; we choose the specific B_{DynTr} such that the predicates obtained after transforming it to an IDT are simple. For $t \in \mathbb{N} \cup \{0\}$, the t^{th} layer of the graph G_{DynTr} is

$$L_t = \{u_0^t, \ldots, u_t^t, v_{\mathrm{tr}(t)}^t\}.$$

The vertices of G_{DynTr} are $V = \cup_{t \in \mathbb{N} \cup \{0\}} L_t$. The source of B_{DynTr} is u_0^0 and the leaves are $U_{\mathrm{leaf}} = \{v_{\mathrm{tr}(t)}^t : t \in \mathbb{N}\}$. For every $t \in \mathbb{N}$ and $0 \leq i \leq t-1$ there are two edges: an edge (u_i^{t-1}, u_i^t) labeled by 1 and an edge (u_i^{t-1}, u_{i+1}^t) labeled by x_t. There are additional edges entering into the leaves: for every $t \in \mathbb{N}$ there are two edges: an edge $(u_{\mathrm{tr}(t)}^{t-1}, v_{\mathrm{tr}(t)}^t)$ labeled by 1 and an edge $(u_{\mathrm{tr}(t)-1}^{t-1}, v_{\mathrm{tr}(t)}^t)$ labeled by x_t.

Informally, the LIBP counts the number of ones in an input σ. Formally, let $\sigma \in \{0,1\}^t$ be an input. Then, there is a path from the source vertex u_0^0 to a vertex u_i^t (for some $0 \leq i \leq t$) that is satisfied by the input σ if and only if σ contains at least i ones; furthermore there is a path from the source u_0^0 to a leaf $v_{\mathrm{tr}(t)}^t$ that is satisfied by σ if and only if σ contains at least $\mathrm{tr}(t)$ ones (this is proved by a simple induction). That is, the LIBP B_{DynTr} computes the $\mathrm{tr}(t)$-dynamic-threshold function. The width B_{DynTr} is $w(t) = t + 1$.

Let $h : \mathbb{N} \to \mathbb{N}$ be any increasing function. We next describe a GIDT T_{DynThr} – the output of the transformation described in Construction 3.11 given B_{DynThr} and h. The i^{th} layer of T_{DynThr} is

$$\{u_{0,j_1,\ldots,j_i} : 0 \leq j_1 \leq h(1), \ldots, 0 \leq j_i \leq h(i)\} \cup \{v_{0,j_1,\ldots,j_{i-1}} : 0 \leq j_1 \leq h(1), \ldots, 0 \leq j_{i-1} \leq h(i-1)\}.$$

In B_{DynThr}, an assignment $\sigma = \sigma_{h(i-1)+1}, \ldots, \sigma_{h(i)}$ satisfies a path from $u_{j_{i-1}}^{h(i-1)}$ to $u_{j_i}^{h(i)}$ if and only if σ contains at least $j_i - j_{i-1}$ ones. Thus, in T_{DynThr} there is an edge $(u_{0,j_1,\ldots j_{i-1}}, u_{0,j_1,\ldots,j_{i-1},j_i})$ labeled by the threshold function $\mathrm{Thr}_{j_i - j_{i-1}}(x_{h(i-1)+1}, \ldots, x_{h(i)})$ (where Thr_b is the b-threshold function, that is, $\mathrm{Thr}_b(y_1, \ldots, y_m) = 1$ if and only if $\sum_{j=1}^{m} y_j \geq b$). Furthermore, an assignment $\sigma = \sigma_{h(i-1)+1}, \ldots, \sigma_t$ satisfies a path from a vertex $u_{j_{i-1}}^{h(i-1)}$ to a leaf $v_{g(t)}^{t}$ (where $h(i-1)+1 \leq t \leq h(i)$ and $\mathrm{tr}(t) - t + h(i-1) \leq j_{i-1} \leq \mathrm{tr}(t)$) if and only if σ contains at least $\mathrm{tr}(t) - j_{i-1}$ ones. Thus, in T_{DynThr} there is an edge $(u_{0,j_1,\ldots j_{i-1}}, v_{0,j_1,\ldots j_{i-1}})$ labeled by

$$\bigvee_{h(i-1)+1 \leq t \leq h(t)} \mathrm{Thr}_{\mathrm{tr}(t)-j_{i-1}}(x_{h(i-1)+1}, \ldots, x_t).$$

Informally, this label is satisfied in T_{DynThr} if in B_{DynThr} at least one path from $u_{0,j_1,\ldots j_{i-1}}$ to a leaf in the i-the generation is satisfied. By Claim 3.12, the GIDT T_{DynThr} computes the $\mathrm{tr}(t)$-dynamic-threshold function.

We implement T_{DynThr} using the secret-sharing scheme of Π_{GIDT}, where each threshold function is implemented using Shamir's t-out-of-n secret-sharing scheme with share size $\log(n+1)$. We next analyze the share size in this scheme for a party p_t in generation i, i.e., $h(i-1)+1 \leq t \leq h(i)$. First, p_t participates in one secret-sharing scheme for each edge $(u_{0,j_1,\ldots,j_{i-1}}, u_{0,j_1,\ldots,j_i})$. There are $\prod_{1 \leq j \leq i}(h(j)+1)$ such edges and the share size of Shamir's scheme realizing the label of each edge is $\log(h(i) - h(i-1)) \leq \log(h(i))$. Second, p_t participates in one secret-sharing scheme for each edge $(u_{0,j_1,\ldots,j_{i-1}}, v_{0,j_1,\ldots,j_{i-1}})$. There are $\prod_{1 \leq j \leq i-1}(h(j)+1)$ such edges. Realizing the label of each edge requires $h(i) - h(i-1)$ applications of Shamir's secret-sharing scheme, resulting in share size $\sum_{t=h(i-1)+1}^{h(i)} \log(t - h(i-1)) < h(i)\log(h(i))$ per edge. To conclude, the share size p_t is $\log(h(i)) \prod_{1 \leq j \leq i}(h(j)+1)$.

To complete the analysis of the share size, we need to choose the function h and compute the share size as a function of t. This was done in [31], taking $h(i) = 2^{2^i} - 1$. Thus, the share size is smaller than

$$2^i \cdot \prod_{1 \leq j \leq i} 2^{2^j} = 2^i \cdot 2^{\sum_{j=1}^{i} 2^j} < 2^i \cdot 2^{2^{i+1}}.$$

As t is in generation i, it must be that $t \geq h(i-1)+1 = 2^{2^{i-1}}$ and the share size is $O(t^4 \log t)$ (as $2^{2^{i+1}} = 2^{4 \cdot 2^{i-1}} = \left(2^{2^{i-1}}\right)^4 \leq t^4$).

Improved Evolving Secret-Sharing Schemes for Small $\mathrm{tr}(t)$. Xing and Yuan's result [41] showed that $\mathrm{tr}(t)$-dynamic-threshold access structures when $\mathrm{tr}(t) = t^\beta$ some for $0 < \beta < 1$ can be realized by an evolving secret-sharing scheme with share size $O(t^{4\beta})$. We show that we can get a similar result using our framework; we get a secret-sharing scheme with share size $O(t^{4\beta}\log t)$ for t^β-dynamic-threshold access structures. The main observation is that in the LIBP B_{DynThr} for $\mathrm{tr}(t) = t^\beta$ we can reduce the number of vertices in layers $h(1), h(2), \ldots$ – we take

$$L_{h(i)} = \{u_0^{h(i)}, \ldots, u_{h(i)^\beta - 1}^{h(i)}\} \cup \{v_{h(i)^\beta}^{h(i)}\}.$$

That is, if the number of ones in a prefix of length $h(i)$ of an input σ is at least $\mathrm{tr}(h(i)) = h(i)^\beta$, then the LIBP can accept σ without looking at bits beyond this input. We do not change other layers, thus the labels on the edges in the GIDT resulting from applying the transformation of

Construction 3.11 to the optimized LIBP remains the same threshold functions. Again we take $h(i) = 2^{2^i} - 1$ and get share size

$$\log(h(i)) \prod_{j=1}^{h(i)} h(i)^{\beta} = 2^i \cdot \prod_{1 \le j \le i} 2^{\beta \cdot 2^j} = 2^i \cdot 2^{\beta \cdot \sum_{j=1}^{i} 2^j} < 2^i \cdot 2^{\beta \cdot 2^{i+1}}.$$

As t is in generation i, it must be that $t \ge h(i-1)+1 = 2^{2^{i-1}}$ and the share size is $O(t^{4\beta} \log t)$ (as $2^{\beta \cdot 2^{i+1}} = 2^{4\beta \cdot 2^{i-1}} = \left(2^{2^{i-1}}\right)^{4\beta}$).

3.3 Evolving Secret-Sharing Schemes for LIBPs with Bounded Width

In this section we present the main application of our construction of evolving secret-sharing schemes for LIBPs, showing that LIBPs with small width can be realized with small share size. We present two results for LIBPs with small width:

- A construction presented in Theorem 3.15 showing that if the width is at most $2^{0.15t}$, then the LIBP can be realized with non-trivial share size of $O(2^{0.97t})$ and if the width is $2^{\varepsilon t}$ for $\varepsilon < 0.004$ the share size is $2^{2\sqrt{\varepsilon} \cdot t}$. This is in contrast with the lower bound of [35], proving that there exists an evolving access structure such that in every evolving secret-sharing realizing it the share size of infinitely many parties p_t is at most $2^{t - o(t)}$.
- A construction presented in Theorem 3.17 that LIBPs with width at most $w(t)$ can be realized by an evolving secret-sharing scheme with share size $(w(2t))^{O(\log t)}$; for example, if $w(t) = t^c$, i.e., the width is polynomial, then the share size of p_t is quasi-polynomial. The second construction achieves smaller share size than the first construction when $w(t) \ll 2^{t/\log^2 t}$. As an application of Theorem 3.17, we show in Section 3.3.1 an evolving secret-sharing scheme for the evolving directed layered st-connectivity access structure, where for every $t \in \mathbb{N}$ the share size of p_t is at most $t^{O(\log t)}$.

Theorem 3.15. *Every LIBP B of width $w(t) \le 2^{0.15t}$ can be realized by an evolving secret-sharing scheme with share size $2^{0.97t}$. Furthermore, for $\varepsilon(t) < 0.04$, every LIBP B of width $w(t) \le 2^{\varepsilon t}$ can be realized by an evolving secret-sharing scheme with share size $2^{2\sqrt{\varepsilon(t)} \cdot t}$.*[5]

Proof. Let $\varepsilon = \varepsilon(t)$, and let $c = c(t) > 1$ to be determined later. We apply Theorem 3.13 to B of width at most $2^{\varepsilon t}$ and $h(i) = c^i$. To realize the predicates of the edges, we use the best known secret-sharing scheme for arbitrary n-party access structures of [5]. Fix a party p_t and let i be the generation of p_t, that is, $c^{i-1} + 1 \le t \le c^i$, in particular

$$c^i \le tc. \tag{1}$$

By Theorem 3.13, the number of shares of secret-sharing schemes of predicates that p_t holds is

$$\begin{aligned} \left(\prod_{1 \le j \le i-1} w(h(j)) \right) (w(h(i)) + 1) = O\left(\prod_{1 \le j \le i} 2^{\varepsilon c^j} \right) = O\left(2^{\varepsilon \sum_{1 \le j \le i} c^j} \right) \\ \le O\left(2^{\varepsilon c^{i+1}/(c-1)} \right) \le O\left(2^{\varepsilon c^2 t/(c-1)} \right), \end{aligned} \tag{2}$$

[5] The constants in the theorem are not optimized.

where the last inequality is from (1). The share size of the secret-sharing scheme of [5] for an n-party access structure is $2^{0.585n}$; we apply it with the parties of a generation i – there are $h(i) - h(i-1) = c^i - c^{i-1} \leq tc - t = (c-1)t$ and the share size for each invocation of the secret-sharing of [5] is $2^{0.585(c-1)t}$. Thus, the size of the share of p_t is

$$O\left(2^{\varepsilon c^2 t/(c-1)} \cdot 2^{0.585(c-1)t}\right) = O\left(2^{\left(\frac{\varepsilon c^2}{c-1}+0.585(c-1)\right)t}\right) = O\left(2^{\left(\varepsilon(c+1)+\frac{\varepsilon}{c-1}+0.585(c-1)\right)t}\right). \quad (3)$$

Taking $\varepsilon = 0.15$ and $c = 1.46$, we obtain from (3) that for every LIBP of width $w(t) \leq 2^{0.15t}$ the share size of p_t is less than $2^{0.97t}$. For the second item of the theorem, take $c = 1 + \sqrt{\frac{\varepsilon}{0.585}}$; for $\varepsilon < 0.04$ we get that $c < 1.27$ and the share size we obtain from (3) is

$$O\left(2^{\left(2.27\varepsilon+2\sqrt{0.585\varepsilon}\right)t}\right) = 2^{2\sqrt{\varepsilon}\cdot t}. \quad (4)$$

□

For LIBPs of width $2^{\varepsilon t}$ for $\varepsilon < 0.04$, the share size decreases as the width decreases. When the width is $w(t) \leq 2^{o(t/\log^2 t)}$, we can get better share size by using special-purpose secret-sharing schemes to realize the labels of the edges. Recall in Construction 3.11, the label on each edge is $\text{reach}_{u,v}$ (or a conjunction of such predicates). When the width of the LIBP is somewhat small, we can use the formula-based secret-sharing scheme of [14] to realize $\text{reach}_{u,v}$; in the secret-sharing scheme of [14] the total share size is the number of leaves of the monotone formula. For completeness, we next describe a monotone Boolean formula for $\text{reach}_{u,v}$.

Claim 3.16. *For every layered finite non-deterministic program $B_{\text{finite}} = (G, u_0, \mu)$ of width at most w and ℓ layers, there exists a monotone Boolean formula with at most $(2w)^{1+\log \ell}$ leaves.*

Proof. The construction of the monotone Boolean formula is as follows, where w.l.o.g., we assume that $\ell+1$ is a power of 2 (this can at most double ℓ); we number the layers of G by layer 0 to layer $\ell-1$. We construct the formula $F_{u,v}$ is recursively. If $\ell = 2$ (i.e., u and v are in consecutive layers), then if $(u,v) \notin E$ then $F_{u,v} = 0$. Otherwise $F_{u,v}$ is the label of the edge (u,v), i.e., either $F_{u,v} = 1$ or $F_{u,v} = x_i$ for some variable x_i. If $\ell > 2$, let $u_1, \ldots, v_w$ be the vertices in G in layer $(\ell-1)/2$; any path from u to v in G passes via exactly one vertex u_i in layer $(\ell-1)/2$, hence

$$F_{u,v} = \bigvee_{i=1}^{w} (F_{u,u_i} \wedge F_{u_i,v}),$$

where F_{u,u_i} and $F_{u_i,v}$ were defined by the recursion.

We next compute the number of leaves in $F_{u,v}$. If $\ell = 2$ the number of leaves is at most 1. If $\ell > 2$, the formula contains $2w$ formulas for graphs with $(\ell-1)/2+1$ layers; by induction, the number of leaves in $F_{u,v}$ is at most $(2w)^{\log \ell}$. □

Theorem 3.17. *Let $w : \mathbb{N} \to \mathbb{N}$ be a non-decreasing function. Every LIBP B of width $w(t)$ can be realized by evolving secret-sharing scheme with share size $(w(2t))^{O(\log(t))}$.*

Proof. We apply Theorem 3.13 to B and $h(i) = 2^i$. To realize the predicates labeling the edges we use the formula-based secret-sharing scheme of [14] with the formula for $\text{reach}_{u,v}$ described in

Claim 3.16. Observe that the predicate $\bigvee_{v \text{ is a leaf in layers } h(i-1)+1,\ldots,h(i) \text{ in } B} \text{reach}_{u,v}$ can be transformed to one instance of $\text{reach}_{u,v'}$ by adding a new vertex v' and connection each leaf v to v' by adding one path to the graph labeled by 1.

Fix a party p_t and let i be the generation of p_t, that is, $2^{i-1}+1 \leq t \leq 2^i$, in particular

$$2^i \leq 2t. \tag{5}$$

The number of shares of secret-sharing schemes for $\text{reach}_{u,v}$ that p_t holds is at most

$$\left(\prod_{1\leq j\leq i-1} w(h(j))\right)(w(h(i))+1) \leq O\left(\prod_{1\leq j\leq i} w(2t)\right) \leq (w(2t))^{O(\log t)}, \tag{6}$$

where the inequalities are obtained from (1) and the monotinicity of h. The share size of the secret-sharing scheme of [14] for the monotone formula for $\text{reach}_{u,v}$ for the graph with $h(i)-h(i-1) \leq h(i)$ layers and width at most $w(h(i)) \leq w(2t)$ is

$$(w(2t))^{O(\log h(i))} = (w(2t))^{O(\log(2^i))} = (w(2t))^{O(\log t)}.$$

Thus, the size of the share of p_t is $(w(2t))^{O(\log t)} \cdot (w(2t))^{O(\log t)} = (w(2t))^{O(\log t)}$.

□

3.3.1 Evolving Secret-Sharing Scheme for Evolving Directed Layered st-Connectivity

In this section, we present another application of our construction of evolving secret-sharing schemes for LIBPs. We construct an LIBP that describes the evolving directed *layered* st-connectivity access structure. Our results show that this evolving access structure can be realized with a share size that is at most polynomially larger than the share size of the finite directed st-connectivity access structure for finite layered graphs. That is, our scheme is comparable to the scheme of Benaloh and Leichter [14].

Before stating our result, we first formally define the directed layered st-connectivity evolving access structure.

Definition 3.18 (Evolving Directed Layered st-Connectivity Access Structures)**.** *An evolving directed layered st-connectivity access structure is defined as follows. The parties in the access structure are the edges of a directed graph $G = (V, E)$, where V is countably infinite, with two fixed vertices $u_s, u_t \in V$, called the source vertex and the target vertex, respectively. The graph is layered, namely, the set V can be partitioned into finite sets $(L_i)_{i\in\mathbb{N}}$ such that $L_0 = \{u_s\}$, and there are edges only from layer i to $L_{i+1} \cup \{u_t\}$, for all i. The order of arrival is such that if a party from layer i arrives, for some i, then no party from a previous layer will arrive (with the order in each layer being arbitrary). A finite set of parties (i.e., edges) is authorized if and only if they contain a directed path from u_s to u_t.*

We prove the following result.

Theorem 3.19. *Let Γ be an evolving directed layered st-connectivity access structure. Then there exists an evolving secret-sharing scheme realizing Γ, such that for all $t \in \mathbb{N}$ the share size of p_t is at most $t^{O(\log t)}$.*

Proof. Let $G = (V, E)$ denote the directed layered graph associated with Γ. Denote its i^{th} layer by L_i^G. We denote the vertices of the graph G by u_i, where $i \in \mathbb{N} \cup \{s, t\}$. We also let e_j denote the edge associated with the j^{th} party in Γ. We first construct an LIBP $B = (G_{\text{BP}} = (V_{\text{BP}}, E_{\text{BP}}), v_{s,0}, \mu)$ for Γ. The idea is that every layer of the LIBP remembers all vertices reachable from u_s via the edges that arrived so far, by order specified in Γ. Moving from one layer to the next, we copy the previous layer, as well as add the potentially new vertex reachable due to considering the new graph edge.

Formally, we define the LIBP as follows. Let $L_0^{\text{BP}} = \{v_{s,0}\}$ denote layer 0. For every $j \geq 1$ we define the j^{th} layer of G_{BP} to contain a copy of the target vertex u_t, a copy of each vertex from any previous layer, and a copy of the endpoint of the edge associated with the j^{th} party. That is, the set of vertices in the j^{th} layer is

$$L_j^{\text{BP}} = \left\{ v_{i,j} : i \in \mathbb{N} \cup \{s,t\} \wedge u_i \in \bigcup_{j'=0}^{j} L_{j'}^G \cup \{u_t\} \right\} \cup \left\{ v_{i,j} : \exists i' \in \mathbb{N} \cup \{s\} \text{ s.t. } e_j = (u_{i'}, u_i) \right\}.$$

There is an edge between any two copies of the same vertex, and for every j, between the two copies of vertices of the j^{th} edge e_j from the consecutive layers L_{j-1}^{BP} and L_j^{BP}, i.e.,

$$E_{\text{BP}} = \{(v_{i,j}, v_{i,j+1}) : i, j \in \mathbb{N}\} \cup \left\{(v_{i',j-1}, v_{i,j}) : (u_{i'}, u_i) = e_j\right\}.$$

Finally, we label each edge as follows. For every i, i' and j we let $\mu((v_{i,j}, v_{i,j+1})) = 1$ and $\mu((v_{i,j}, v_{i',j+1})) = (u_i, u_{i'})$. We now prove that B computes the evolving access structure Γ. This follows from the following two claims.

Claim 3.20. *Fix $i \in \mathbb{N} \cup \{s, t\}$. Assume that the parties that arrived contain a finite simple path in G from u_s to the vertex u_i. Then there exists a path in G_{BP} from $v_{0,0}$ to $v_{i,j}$ for some $j \in \mathbb{N}$, such that all the edges along the path are labeled 1.*

Proof. The proof is done by induction on the length of the path. The claim clearly holds for a path of length 0, since the only choice for i is to be s. Next, assume that the claim holds for all paths of length ℓ. We prove it for any path of length $\ell + 1$. Let $u_{i'}$ denote the penultimate vertex in the path, namely, the set of parties that arrived contains a finite simple path in G from u_s to the vertex $u_{i'}$, and the party associated with the edge $(u_{i'}, u_i)$ arrived. Let $e_j = (u_{i'}, u_i)$.

By the induction hypothesis, there exists a path in G_{BP} from $v_{0,0}$ to $v_{i',k}$ for some $k \in \mathbb{N}$, such that all the edges along the path are labeled 1. Since G is layered it holds that $j > k$, hence by construction there is a path from $v_{i',k}$ to $v_{i',j-1}$ whose edges are labeled 1. Finally, since the party associated with the edge $(u_{i'}, u_i) = e_j$ arrived, it follows that the edge $(v_{i',j-1}, v_{i,j})$ is labeled 1. □

Claim 3.21. *Fix $i \in \mathbb{N} \cup \{s, t\}$. Assume that the parties that arrived are such that the LIBP contains a finite path from $v_{s,0}$ to $v_{i,j}$ for some $j \in \mathbb{N}$, where all edges are labeled 1. Then the set of parties contains a finite path in G from u_s to u_i.*

Proof. The proof is done by induction on the length of the path. For a path of length 0, the claim trivially holds. Assume that the claim holds for all paths of length ℓ. We prove the claim for all paths of length $\ell + 1$. Let $v_{i',\ell}$ denote the penultimate vertex in the path, that is, there is a path from $v_{s,0}$ to $v_{i',\ell}$ where all edges are labeled 1, and the edge $(v_{i',\ell}, v_{i,\ell+1})$ has label 1. By the induction hypothesis, there is a finite path in G from u_s to $u_{i'}$. Then if $i' = i$ the claim follows. Otherwise, $(u_{i'}, u_i) \in E$ by the definition of the LIBP. □

The correctness of the LIBP follows from the above two claims by taking $i = t$. To complete the proof of the theorem, we next analyze the width of B and apply Theorem 3.17. Notice that for each layer we add at most 2 new vertices, namely a copy of u_t and a copy of the endpoint of the j^{th} party to arrive. Thus the width of layer j is at most $j + 2$. By Theorem 3.17, this implies that B can be realized with share size at most $(2j + 2)^{O(\log j)}$. □

Remark 3.22 (On the importance of the ordering of the parties). We next explain why our scheme fails for general st-connectivity evolving access structures, where the order of arrival is not consistent with the layers. Indeed, we could miss a path from u_s to u_t in G due to the fact that an edge came too late. For example, consider the path (u_s, u_1, u_2, u_t). If the edges are ordered (u_s, u_1), (u_2, u_t), (u_1, u_2), then in the LIBP we constructed we have copies of $\{u_s, u_1\}$ as both layers 1 and 2. A copy of the vertex u_2 only appears in the third layer after we are done processing all edges.

4 Improving the Share Size for Dynamic Threshold Access Structures with Large Threshold

In this section, we consider the dynamic-threshold access structure with a large threshold. In more detail, we consider a threshold $\mathrm{tr}(t) \geq t - t^{\beta}$, where $\beta \in (0, 1)$ is a constant. We construct an evolving secret-sharing scheme for the $\mathrm{tr}(\cdot)$-dynamic-threshold access structure with better share size than what is given in [31]. The main result of this section is the following.

Theorem 4.1. *Let $\beta \in (0, 1)$ be a constant, and let Γ be the evolving $\mathrm{tr}(\cdot)$-dynamic-threshold access structure, where $\mathrm{tr}(t) \geq t - t^{\beta}$. Then there exists an evolving secret-sharing scheme realizing Γ, such that for all sufficiently large t, the share size of p_t is at most $O(t^{1+2\sqrt{\beta}+\beta} \cdot \log t)$.*

Note that the share size in Theorem 4.1 is better than the share size in the scheme of [31] for every $\beta < 1$. The rest of the section is organized as follows. In Section 4.1, we introduce a secret-sharing scheme for the finite version of the dynamic-threshold access structure, which we refer to as a finite dynamic-threshold access structure. In Section 4.2, we show the construction by Xing and Yuan [41], which reduced the task of constructing evolving dynamic secret-sharing schemes to the task of constructing schemes for *finite* dynamic-threshold access structures. In Section 4.3, we combine the above two results and prove Theorem 4.1.

4.1 Finite Dynamic-Threshold Access Structures

We first formally define finite-dynamic-threshold access structures. We define and state our results for it over a set of parties whose index is shifted by a number n_0.

Definition 4.2 (Finite Dynamic Access Structure)**.** *We define the $(\mathrm{tr}(\cdot), n_0, n)$-finite-dynamic-threshold access structure, denote $\Gamma_{n_0,n,\mathrm{tr}(\cdot)}$, to be the finite access structure whose parties are $\{p_{n_0+1}, \ldots, p_{n_0+n}\}$, and a set A of parties is authorized if for some $n_0 + 1 \leq t \leq n_0 + n$ it holds that $|A \cap \{p_{n_0+1}, \ldots, p_t\}| \geq \mathrm{tr}(t)$.*

While in the rest of this work, we consider secret-sharing schemes for single-bit secrets, we will now consider a generalization to secret-sharing schemes for ℓ-bit secrets (as we will need it for our construction for evolving secret-sharing schemes). We construct a secret-sharing scheme for any finite dynamic-threshold access structure. We will use the following two simple facts.

Fact 4.3. *Let* $\mathrm{tr} : \mathbb{N} \to \mathbb{N}$ *and let* Γ *be a finite/evolving* $\mathrm{tr}(\cdot)$*-dynamic-threshold access structure, whose set of parties are* $\{p_{n_0+i}\}_{i\in A}$*, where either* $A = [n]$ *for some* $n \in \mathbb{N}$ *or* $A = \mathbb{N}$*. If* $t \in \mathbb{N}$ *is such that* $\mathrm{tr}(t) - \mathrm{tr}(t-1) \geq 1$*, then there is no minimal authorized set* A *whose maximal party is* p_t*.*

Proof. Assume towards contradiction there exists a minimal $A \in \Gamma$ and $t \in \mathbb{N}$, such that the maximal party in A is p_t. Then $|A \cap \{p_{n_0+1}, \ldots, p_t\}| \geq \mathrm{tr}(t)$, hence

$$|A \cap \{p_{n_0+1}, \ldots, p_{t-1}\}| \geq |A \cap \{p_{n_0+1}, \ldots, p_t\}| - 1 \geq \mathrm{tr}(t) - 1 \geq \mathrm{tr}(t-1),$$

contradicting the minimality of A. □

Fact 4.4. *Let* $\mathrm{tr} : \mathbb{N} \to \mathbb{N}$ *and let* Γ *be a finite/evolving* $\mathrm{tr}(\cdot)$*-dynamic-threshold access structure. Then there exists* $\mathrm{tr}' : \mathbb{N} \to \mathbb{N}$ *such that* $1 \leq \mathrm{tr}'(t) \leq t+1$ *and* $0 \leq \mathrm{tr}'(t+1) - \mathrm{tr}'(t) \leq 1$ *for all* $t \in \mathbb{N}$*, and the* $\mathrm{tr}'(\cdot)$*-dynamic-threshold access structure equals* Γ*.*

Proof. Let $\mathrm{tr}'(\cdot)$ be a minimal such function, i.e., for every function $\mathrm{tr}'' : \mathbb{N} \to \mathbb{N}$ such that $\mathrm{tr}''(t) \leq \mathrm{tr}'(t)$ for all $t \in \mathbb{N}$ and $\mathrm{tr}''(\cdot)$-dynamic-threshold access structure equals Γ, it holds that the $\mathrm{tr}''(t) = \mathrm{tr}'(t)$ for all $t \in \mathbb{N}$. The minimality of $\mathrm{tr}'(\cdot)$ clearly implies that $1 \leq \mathrm{tr}'(t) \leq t+1$ for all t. Assume towards contradiction that there exists $t \in \mathbb{N}$ such that $\mathrm{tr}'(t) \geq \mathrm{tr}'(t-1) + 2$. Then by Fact 4.3 there is no minimal authorized set A whose maximal party is p_t. Consider the function $\mathrm{tr}'' : \mathbb{N} \to \mathbb{N}$ defined as $\mathrm{tr}''(t) = \mathrm{tr}'(t) - 1$ and $\mathrm{tr}''(i) = \mathrm{tr}'(i)$ for all $i \neq t$. By Fact 4.3, since $\mathrm{tr}''(t) - \mathrm{tr}''(t-1) \geq 1$, also the $\mathrm{tr}''(\cdot)$-dynamic threshold access does not contain any minimal authorized set whose maximal party is p_T. Then the $\mathrm{tr}''(\cdot)$-dynamic-threshold access structure equals Γ. However, this contradicts the minimality of $\mathrm{tr}'(\cdot)$. □

A useful tool towards proving Theorem 4.1 is an implementation of the $(\mathrm{tr}(\cdot), n)$-finite-dynamic-threshold access structure (for ℓ-bit secrets) for a given function $\mathrm{tr}(\cdot)$, with a small share size. Theorem 2.5 could be naively used to construct such schemes with share size $n \cdot \max\{\ell, \lceil \log(n+1) \rceil\}$. In particular, this can be achieved by using a separate Shamir scheme for each party $i \in [n]$, that is,

$$\Gamma_{n_0,n,\mathrm{tr}(\cdot)} = \bigvee_{t=1}^{n} \mathrm{TR}_{t,\mathrm{tr}(t)},$$

where $\mathrm{TR}_{t,\mathrm{tr}(t)}$ is the $\mathrm{tr}(t)$-out-of-t threshold access structure. However, as we argue in the following claim, this upper bound can be substantially improved when $\mathrm{tr}(n) \geq n - n^{\beta}$. To construct such a scheme, we begin with the above naive construction and show how to omit many redundant Shamir scheme instantiations.

Claim 4.5. *Fix constants* $\beta \in (0,1)$ *and* $m, n_0, n \in \mathbb{N}$ *such that* $m \leq n_0$*. Let* $\mathrm{tr}' : \{n_0+1, \ldots, n_0+n\} \to \mathbb{N}$ *such that* $\mathrm{tr}'(t) \geq t - t^{\beta}$ *for all* $t \in \{n_0+1, \ldots, n_0+n\}$*. Let* $\mathrm{tr}(t) = \mathrm{tr}'(t) - m$ *for all* t*. Then, for every* $\ell \in \mathbb{N}$*, there exists a secret-sharing scheme realizing* $\Gamma_{n_0,n,\mathrm{tr}(\cdot)}$ *with* ℓ*-bit secrets, in which the share size of every party is at most* $(n_0+n)^{\beta} \cdot \max\{\ell, \lceil \log(n+n_0+1) \rceil\}$*.*

Proof. We begin our construction with the naive scheme, where for every i we share the secret between the first i parties using a $\mathrm{tr}(i)$-out-of-i Shamir's secret-sharing scheme. Now, by Fact 4.3, it follows that if $\mathrm{tr}(i-1) < \mathrm{tr}(i)$ then there is no minimal authorized set whose maximal party is p_i. Therefore, there is no need for the i^{th} Shamir's scheme in this case. Thus, it suffices to use

Shamir's scheme only for each i such that $\mathrm{tr}(i) = \mathrm{tr}(i-1)$.[6] The total number of Shamir's scheme used is at most n^β. Indeed, since $\mathrm{tr}(\cdot)$ starts at 0, ends at $\geq n - n^\beta$ after n steps, and by Fact 4.4 grows by at most 1 in each step, there are at most n^β indexes i such that $\mathrm{tr}(i) = \mathrm{tr}(i-1)$.

We next present the formal construction. By Fact 4.4 we may assume without loss of generality that $0 \leq \mathrm{tr}(t+1) - \mathrm{tr}(t) \leq 1$ for all $t \in \mathbb{N}$. We further define $\mathrm{tr}(n_0) = \mathrm{tr}(n_0+1) - 1$.

Construction 4.6 (A Secret-Sharing Scheme $\Pi_{\mathsf{fin}}(n_0, n)$ for a Finite Dynamic Access Structure)**.**
Input: $s \in \{0,1\}^\ell$.
The sharing algorithm:

1. For every $i \in \{1, \ldots, n\}$ such that $\mathrm{tr}(i) = \mathrm{tr}(i-1)$ do the following.
   (a) Share s using a $\mathrm{tr}(i)$-out-of-i Shamir's secret-sharing scheme over the field $\mathbb{F}_{2^{\max\{\ell, \lceil \log(i+1) \rceil\}}}$ (see Theorem 2.5).
   (b) For every $j \in [i]$, let $\mathsf{sh}_{j,i}$ denote the j^{th} share in the above scheme.
2. For every $j \in [n]$, the share sh_{n_0+j} is $(\mathsf{sh}_{j,i})_{j \leq i \leq n : \mathrm{tr}(i) < \mathrm{tr}(i+1)}$.

We first prove the correctness of the scheme. Let $A \in \Gamma$ be a minimal authorized set. We show that the parties in A can reconstruct the secret s. Let $n_0 + 1 \leq t \leq n_0 + n$ be the maximal index such that $p_t \in A$. Then $|A \cap \{p_{n_0+1}, \ldots, p_t\}| \geq \mathrm{tr}(t)$. By Fact 4.3, it follows that $\mathrm{tr}(t-1) = \mathrm{tr}(t)$, as otherwise there is no minimal authorized set whose maximal party is p_t. Therefore, the parties in A hold at least $\mathrm{tr}(t)$ shares in the $\mathrm{tr}(t)$-out-of-t scheme for s.

We now prove the security of the scheme. Let $A \notin \Gamma$ be an unauthorized set. That is, for any $i \in [n]$ such that $p_{n_0+i} \in A$, it holds that $|\{p_{n_0+j} : j \leq i\}| < \mathrm{tr}(i)$. Therefore, the shares given to the parties in A at iteration i are $(\mathsf{sh}_{j,\mathrm{tr}(i)})_{j \leq i, p_j \in A}$. Since there are less than $\mathrm{tr}(i)$ such shares, and since the shares computed in all iterations are independent, it follows that the set A learns no information on s.

We next analyze the share complexity. It holds that

$$n - n^\beta \leq \mathrm{tr}(n) = \sum_{i=1}^{n} (\mathrm{tr}(i) - \mathrm{tr}(i-1)) = |\{i : \mathrm{tr}(i) > \mathrm{tr}(i-1)\}| = n - |\{i : \mathrm{tr}(i) = \mathrm{tr}(i-1)\}|.$$

Thus, $|\{i : \mathrm{tr}(i+n_0) = \mathrm{tr}(i+n_0-1)\}| \leq (n_0+n)^\beta$. Thus, we execute Shamir's secret-sharing scheme at most $(n_0+n)^\beta$ times. Therefore, the share of each party is at most $(n_0+n)^\beta \cdot \max\{\ell, \lceil \log(n + n_0 + 1) \rceil\}$. □

4.2 The Evolving Secret Sharing of Xing and Yuan

The next lemma states that we can reduce the task of sharing for the evolving dynamic-threshold access structure to the task of sharing for the finite dynamic-threshold access structure. The scheme we construct is the same as the scheme constructed by Xing and Yuan [41], using a different formulation. The correctness and security of the scheme works for any threshold function.

Let us first explain the intuition of the sharing algorithm. Similarly to previous schemes, we partition the parties into generations of increasing size. Let g_i denote the size of the i^{th} generation.

[6] It is also possible remove the use of Shamir's scheme for $i-1$ in this case as well without affecting correctness. However, this will not affect the share size of the resulting scheme.

First, we share the secret s among the first generation using the (finite) secret-sharing scheme Π_{fin} given in Construction 4.6. Second, we handle the case where there is a minimal authorized set containing parties from future generations as follows. We share s using a $\mathrm{tr}(g_1)$-out-of-$(g_1+\mathrm{tr}(g_1))$ Shamir's secret-sharing scheme, giving one share to each party in the generation. This results with $\mathrm{tr}(g_1)$ "extra shares". These "extra shares" are then shared recursively among the parties from the second generation onward, using an appropriate (dynamic) threshold function.

To see how to choose the threshold function, consider for example the case where $\mathrm{tr}(g_1)-1$ parties from the first generation arrived. If at time t the additional number of parties to arrive is at least $\mathrm{tr}(t)-\mathrm{tr}(g_1)+1$, then they should be able to reconstruct the secret. We share the first "extra shares" among the parties in all future generations using an evolving dynamic scheme with threshold $\mathrm{tr}(t)-\mathrm{tr}(g_1)+1$. Then, the parties can reconstruct this "extra share" and use the additional $\mathrm{tr}(g_1)-1$ shares from the first generation to reconstruct the secret. The other "extra shares" will be shared using an evolving dynamic scheme with a higher threshold (e.g., the second "extra share" will be shared with dynamic threshold $\mathrm{tr}(t)-\mathrm{tr}(g_1)+2$). Therefore, an authorized set containing k parties from the first generation will be able to reconstruct $\mathrm{tr}(g_1)-k$ "extra shares" and will be able to reconstruct the secret.

Lemma 4.7. *Let $\mathrm{tr}:\mathbb{N}\to\mathbb{N}$ and let Γ be the evolving $\mathrm{tr}(\cdot)$-dynamic-threshold access structure. Suppose that for every $n_0,n,\ell\in\mathbb{N}$ and $\mathrm{tr}':\mathbb{N}\to\mathbb{N}$ there exists a finite secret-sharing scheme $\Pi_{\mathrm{tr}'(\cdot),n_0,n,\ell}$ realizing the $(\mathrm{tr}'(\cdot),n_0,n)$-finite dynamic-threshold access structure for ℓ-bit secrets, with share complexity $\mathsf{size}(\mathrm{tr}'(\cdot),n,\ell)$. Then for every $c\in\mathbb{N}$, there exists an evolving secret-sharing scheme realizing Γ such that for all sufficiently large $t\in\mathbb{N}$, the share size of p_t is at most*

$$t^{\frac{c}{c-1}}\cdot(\mathsf{size}(\mathrm{tr}(\cdot),\ t^c,\ \log t+3)+c\cdot\log t+3).$$

Proof. We now provide a formal description of the construction, using a recursive procedure. We stress that formally, instead of recursively calling the scheme, the dealer maintains many instantiations of the sharing scheme in its head with different parameters (i.e., the secret, which parties participate, and the threshold function). The share of party p_t is the concatenation of all shares that are associated with it in all of the calls.

We first introduce some notations. Let $h:\mathbb{N}\to\mathbb{N}$ be define as $h(0)=0$ and $h(i)=2^{c^i}$ for every $i\in\mathbb{N}\setminus\{0\}$. The i^{th} generation is then defined to be $G_i=\{p_t : h(i-1)+1\le t\le h(i)\}$, and let $g_i=|G_i|$. In particular, $g_i\le h(i)=2^{c^i}$. Finally, by Fact 4.4 we may assume without loss of generality that $1\le\mathrm{tr}(t)\le t+1$ for all $t\in\mathbb{N}$.

Construction 4.8 (A Recursive Secret-Sharing Scheme $\Pi_{\mathrm{rec}}(\ell,s,i,\mathrm{tr}(\cdot))$ for dynamic-threshold)**.**
Input: The size of the secret ℓ, the secret $s\in\{0,1\}^\ell$, a generation number $i\in\mathbb{N}$, and a threshold function $\mathrm{tr}:\mathbb{N}\to\mathbb{N}$.
The sharing algorithm:

- When the first party in G_i (i.e., $p_{h(i-1)+1}$) arrives do:
  - Share s among the parties of G_i using $\Pi_{\mathrm{tr}(\cdot),h(i-1),g_i,\ell}$. Let $\mathsf{sh}^{\mathsf{fin}}_{i,1},\ldots,\mathsf{sh}^{\mathsf{fin}}_{i,g_i}$ denote the resulting shares.
  - Share s using the $\mathrm{tr}(g_i)$-out-of-$(g_i+\mathrm{tr}(g_i))$ Shamir's secret-sharing scheme over the field $\mathbb{F}_{2^{\max\{\ell,\lceil\log(g_i+\mathrm{tr}(g_i)+1)\rceil\}}}$ (see Theorem 2.5). Denote the shares by $\mathsf{sh}^{\mathsf{ex}}_{i,1},\ldots,\mathsf{sh}^{\mathsf{ex}}_{i,g_i+\mathrm{tr}(g_i)}$. We refer to the last $\mathrm{tr}(g_i)$ shares $\mathsf{sh}^{\mathsf{ex}}_{i,g_i+1},\ldots,\mathsf{sh}^{\mathsf{ex}}_{i,g_i+\mathrm{tr}(g_i)}$ as "extra shares".

– When $p_t \in G_i$ arrives, append $(\mathsf{sh}^{\mathsf{fin}}_{i,t}, \mathsf{sh}^{\mathsf{ex}}_{i,t-h(i-1)})$ to the t^{th} share sh_t.

- When the first party in G_{i+1} (i.e., $p_{h(i)+1}$) arrives, for every $j \in \{1, \ldots, \mathrm{tr}(g_i)\}$ share the "extra share" $\mathsf{sh}^{\mathsf{ex}}_{i,g_i+j}$ using

$$\Pi_{\mathrm{rec}}(\max\{\ell, \lceil \log(g_i + \mathrm{tr}(g_i) + 1) \rceil\}, \mathsf{sh}^{\mathsf{ex}}_{i,g_i+j}, i+1, \mathrm{tr}_j(\cdot)),$$

  where $\mathrm{tr}_j(t) = \mathrm{tr}(t) - j + 1$ for all $t \in \mathbb{N}$.

We next analyze the above secret-sharing scheme, i.e., for parameters $\ell = 1$, $i = 1$, and $\mathrm{tr}(\cdot)$. We first prove the correctness of the scheme. Let A be an authorized set, and let $t \in \mathbb{N}$ be the maximal number such that $p_t \in A$. Note that without loss of generality we may assume that A is minimal. Let $i \in \mathbb{N}$ be generation of p_t, i.e., $p_t \in G_i$, and let j be the minimal index such that $A \cap G_j \neq \emptyset$. Note that $A \subseteq G_j \cup \ldots \cup G_i$. We prove by induction on $i - j$ that A can recover the secret. For the base case, where $j = i$, it holds that $A \subseteq G_i$. Therefore, by the correctness of the finite scheme, the parties can reconstruct the secret. Assume that the claim holds for $i - j - 1$. We show that it holds for $i - j$. Let $k = |A \cap G_j|$. Since A is minimal, it follows that $|A| \geq \mathrm{tr}(t)$ and that $k \leq \mathrm{tr}(g_j) - 1$. Therefore, $|A \setminus G_j| = |A| - |A \cap G_j| \geq \mathrm{tr}(t) - k$. Therefore, by the induction hypothesis, the parties in $A \setminus G_j$ can recover $\mathsf{sh}^{\mathsf{ex}}_{j,g_j+k+1}, \ldots, \mathsf{sh}^{\mathsf{ex}}_{j,g_j+\mathrm{tr}(g_j)}$. Thus, they can recover $\mathrm{tr}(g_j) - k$ "extra shares" and they hold k share; by the correctness of Shamir's secret-sharing scheme, the parties in A can recover the secret.

We next show the security of the scheme. Let $A \notin \Gamma$ be an unauthorized set, and let $t \in \mathbb{N}$ be the maximal number such that $p_t \in A$. Similarly to before, let $i \in \mathbb{N}$ be generation of p_t, i.e., $p_t \in G_i$, and let j be the minimal index such that $A \cap G_j \neq \emptyset$. We prove by induction on $i - j$ that A learns nothing about the secret. For the base case, where $i = j$, it holds that $A \subseteq G_i$. Therefore, by the security of the finite secret-sharing scheme and since they hold less than $\mathrm{tr}(g_i)$ shares from Shamir's scheme, it follows that the parties in A gain no information on the secret s. Assume that the claim holds for $i - j - 1$. We prove it for $i - j$. Let $k = |A \cap G_j|$. Since $A \notin \Gamma$, it follows that $|A| \leq \mathrm{tr}(t) - 1$. Then $|A \setminus G_j| = |A| - |A \cap G_j| \leq \mathrm{tr}(t) - k - 1$. Therefore, the parties in $A \setminus G_j$ can recover (a subset of) the shares $\mathsf{sh}^{\mathsf{ex}}_{j,g_j+k+2}, \ldots, \mathsf{sh}^{\mathsf{ex}}_{j,g_j+\mathrm{tr}(g_j)}$. Moreover, by the induction hypothesis, the parties in $A \setminus G_j$ learn nothing on any of the shares $\mathsf{sh}^{\mathsf{ex}}_{j,g_j+1}, \ldots, \mathsf{sh}^{\mathsf{ex}}_{j,g_j+k+1}$. Thus, the parties in A learn at most $\mathrm{tr}(g_j) - 1$ of the shares $\mathsf{sh}^{\mathsf{ex}}_{j,1}, \ldots, \mathsf{sh}^{\mathsf{ex}}_{j,g_j+\mathrm{tr}(g_j)}$, hence by the security of Shamir's scheme, these shares hold no information on s. Moreover, the shares given to $A \cap G_j$ using the finite sharing scheme are independent of the shares given recursively, hence they give no information to the parties in A.

It remains to analyze the share complexity of the scheme. Fix $t \in \mathbb{N}$ and let $i \in \mathbb{N}$ denote the index of the generation G_i such that $p_t \in G_i$. Then $h(i-1) + 1 \leq t \leq h(i)$. This implies that

$$2^{c^{i-1}} < t \leq 2^{c^i} = \left(2^{c^{i-1}}\right)^c < t^c. \tag{7}$$

The share of p_t consists of $\mathsf{sh}^{\mathsf{fin}}_{i,t}$ and $\mathsf{sh}^{\mathsf{ex}}_{i,t-h(i-1)}$ for every recursive call to Π_{rec} with generation number i. Let us first count the number of such recursive calls. Recall that $\mathrm{tr}_j(t) = \mathrm{tr}(t) - j$ for all $j, t \in \mathbb{N}$. Then there is one such call for every sequence $j_1, \ldots, j_{i-1}$ such that $0 \leq j_k \leq \mathrm{tr}_{j_{k-1}}(g_k) - 1$ for all $k \in [i-1]$, where we let $j_0 = 0$. Therefore, for every $k \in [i-1]$ it holds that

$$j_k \leq \mathrm{tr}(g_k) - 1 \leq g_k \leq h(k) = 2^{c^k}.$$

We conclude that the number of such sequences is at most

$$\prod_{k=1}^{i-1} 2^{c^k} = 2^{\sum_{k=1}^{i-1} c^k} \leq 2^{\frac{c^i}{c-1}} = (2^{c^i})^{\frac{1}{c-1}} < t^{\frac{c}{c-1}},$$

where the last inequality is by Equation (7).

Now, for every $k \in [i]$ let ℓ_k denote the size of the secret at iteration k. Then $\ell_1 = 1$ and $\ell_{k+1} = \max\{\ell_k, \lceil \log(g_k + \mathrm{tr}(g_k) + 1) \rceil\} = \lceil \log(g_k + \mathrm{tr}(g_k) + 1) \rceil$ for every $k < i$. Therefore,

$$\ell_i \leq \log(g_{i-1} + \mathrm{tr}(g_{i-1}) + 1) + 1 \leq \log(2g_{i-1} + 2) + 1 \leq \log(h(i-1) + 1) + 2 < \log(t+1) + 2 < \log t + 3,$$

where the second inequality is by the assumption that $\mathrm{tr}(t) \leq t + 1$ for all t. This implies that the size of the share $\mathsf{sh}^{\mathsf{fin}}_{i,t}$ is at most

$$\mathsf{size}(\mathrm{tr}(\cdot),\ g_i,\ \ell_i) \leq \mathsf{size}(\mathrm{tr}(\cdot),\ t^c,\ \log t + 3)$$

for sufficiently large t, and the size of $\mathsf{sh}^{\mathsf{ex}}_{i,t-h(i-1)}$ is at most

$$\log(g_i + \mathrm{tr}(g_i) + 1) + 1 \leq \log(h(i)) + 3 \leq c \cdot \log t + 3,$$

for sufficiently large t. We conclude that the share size of p_t is at most

$$t^{\frac{c}{c-1}} \cdot (\mathsf{size}(\mathrm{tr}(\cdot),\ t^c,\ \log t + 3) + c \cdot \log t + 3).$$

□

4.3 Putting it all Together

We are now ready to prove Theorem 4.1.

Proof. We prove the theorem instantiating Lemma 4.7 with the secret-sharing scheme from Claim 4.5, and choosing the optimal value for c. Consider the secret-sharing scheme $\Pi_{\mathrm{rec}}(\{0,1\}, s, 1, \mathrm{tr}(\cdot))$, instantiated with the (finite) secret-sharing scheme Π_{fin} guaranteed to exist by Claim 4.5. Then the resulting share size is at most

$$t^{\frac{c}{c-1}} \cdot \left(t^{c\beta} \cdot (c \cdot \log t + 1) + c \cdot \log t + 3 \right) \leq 4c \cdot t^{\frac{c}{c-1} + c\beta} \cdot \log t,$$

for all sufficiently large t's. We next minimize the exponent $\frac{c}{c-1} + c\beta = 1 + \frac{1}{c-1} + c\beta$ with respect to c. Taking the derivative results in

$$\beta - \frac{1}{(c-1)^2},$$

which equals to 0 when $c = \beta^{-1/2} + 1$. Therefore, the share complexity of p_t is at most

$$4(1 + \sqrt{\beta}) \cdot t^{1 + 2\sqrt{\beta} + \beta} \cdot \log t.$$

□

5 Evolving Secret-Sharing Scheme for Evolving Slice Access Structures

In this section, we present our construction of evolving 2-slice secret-sharing schemes and 3-slice secret-sharing schemes. The schemes are given in Section 5.2 and in Section 5.3, respectively. CDS protocols, which are used in our schemes, are defined in Section 5.1.

5.1 Conditional Disclosure of Secrets (CDS)

In a k-server CDS protocol, there are k servers holding the same secret s and a common random string. Additionally, the i^{th} server, for $1 \leq i \leq k$, holds a private input x_i. In addition, there is a referee, which knows $x_1, \ldots, x_k$ but, prior to the execution of the protocol, the referee does not know the secret and the common random string. In a CDS protocol, each server sends a single message to the referee. The message of the i^{th} server is a function of the secret, the common random string, and its private input x_i; the message is independent of the other inputs and the messages computed by other servers. The referee should learn the secret s if and only if $\varphi(x_1, \ldots, x_k) = 1$, for a fixed predicate φ.

Definition 5.1 (Conditional Disclosure of Secrets (CDS) Protocols [27])**.** *Let $X_1, \ldots, X_k$ be sets. A k-server* conditional disclosure of secrets *(CDS) protocol with domain of secrets S, domain of common random strings R, and domain of messages $M_1, \ldots M_k$ is a tuple of deterministic algorithms* $\text{ENC}_1, \ldots, \text{ENC}_k$*, and* DEC *with the following syntax.*

- $\text{ENC}_i : S \times X_i \times R \to M_i$*, for all $1 \leq i \leq k$, is the encoding algorithm of the i^{th} server, whose inputs are the secret, the private input, and the common random string. Its output is the message of the i^{th} server.*
- $\text{DEC} : X_1 \times \cdots \times X_k \times M_1 \times \cdots \times M_k \to S$ *is the decoding algorithm of the referee, whose inputs are the private inputs of the servers and the messages sent by them. Its output is a secret.*

For an input $x \in X_1 \times \cdots \times X_k$, a secret s, and common randomness r, we let

$$\text{ENC}(x, s; r) \triangleq (\text{ENC}_1(x_1, s; r), \ldots, \text{ENC}_1(x_k, s; r)).$$

Let $\varphi : X_1 \times \cdots \times X_k \to \{0, 1\}$ be a predicate. We say that a CDS protocol computes the predicate φ if it satisfies the following properties.

Correctness. *For all $x \in X_1 \times \cdots \times X_k$ such that $\varphi(x) = 1$, for all $r \in R$, and for all $s \in S$ it holds that*

$$\text{DEC}(\text{ENC}(x, s; r)) = s.$$

Security. *For every input $x \in X_1 \times \cdots \times X_k$, for which $a(x) = 0$, and every pair of secrets $s_1, s_2 \in S$ it holds that*

$$\text{ENC}(x, s_1; r) \text{ and } \text{ENC}(x, s_2; r) \text{ are identically distributed,}$$

where r is sampled with uniform distribution from R.

The message size of a CDS protocol is defined as the size of the largest message sent by the servers, i.e., $\max_{i\in[k]} \log |M_i|$.

For $N, k \in \mathbb{N}$ we will consider the predicate INDEX_N^k. Here, an index $i \in [N^{k-1}]$ is distributed amongst the first $k-1$ servers, and the k^{th} server holds a string $D \in \{0,1\}^{N^{k-1}}$ viewed as a $(k-1)$-dimensional array, called the database. The output of the predicate is the i^{th} bit of D. Formally, it is defined as follows.

Definition 5.2 (The Predicate INDEX_N^k)**.** *Let $k, N \in \mathbb{N}$, where $k \geq 2$. Define the predicate $\mathrm{INDEX}_N^k : [N]^{k-1} \times \{0,1\}^{N^{k-1}} \to \{0,1\}$ as $\mathrm{INDEX}_N^k(i_1, \ldots, i_{k-1}, D) = D[i_1, \ldots, i_{k-1}]$.*

Liu, Vaikuntanathan, and Wee [34] constructed an efficient CDS protocol for INDEX_N^{k-1} as described in the following theorem.

Theorem 5.3 ([34])**.** *For all $k, N \in \mathbb{N}$ such that $2 \leq k \leq \log N$, there is a k-server CDS protocol for INDEX_N^k whose communication complexity is $2^{O(\sqrt{k \log N} \cdot \log\log N)}$.*

5.2 Scheme for Evolving 2-Slice Access Structures

We show an evolving secret-sharing scheme for evolving 2-slice secret-sharing schemes. We prove the following theorem.

Theorem 5.4. *Let Γ be an evolving 2-slice access structure. Then for every $\varepsilon > 0$, there exists an evolving secret-sharing scheme realizing Γ, such that for all $t \in \mathbb{N}$, the share size of p_t is at most $2^{O\left((\log t)^{\frac{1}{\sqrt{2}}+\varepsilon} \cdot \sqrt{\log\log t}\right)}$.*

Proof. Let us first provide an intuitive description of the secret-sharing scheme. We first show a scheme whose share complexity is slightly worse than what is stated in Theorem 5.4. First, we handle authorized sets of size at least 3 using the evolving 3-threshold scheme of Komargodski et al. [30]. For authorized sets of size exactly 2 we do the following. We partition the parties into generations. Let G_i denote the i^{th} generation. We then use the finite secret-sharing scheme for 2-slice functions described in Theorem 2.9 to share s among the parties of every two consecutive generations, i.e., $G_i \cup G_{i+1}$ for every $i \in \mathbb{N}$. Finally, we need to handle pairs of parties $\{p_{t'}, p_t\} \in \Gamma$, where $t' < t$ and $p_{t'}$ appears at least 2 generations prior to p_t. Here we give $p_{t'}$ a random bit $r_{t'}$. Then, for p_t (which arrives later), we give it $s \oplus r_t$. In the above scheme, the share size of p_t is at most $2^{\tilde{O}((\log t)^{\sqrt[3]{4}/2)}}$. To obtain a scheme with better share complexity, instead of using the scheme of Theorem 2.9 to share s among the parties of every two consecutive generations, we will do this for every k consecutive generations. We show that the share complexity of the scheme improves the larger k is.

We next present the formal construction. We first introduce some notations. Let Π_3 denote the evolving secret-sharing scheme for the evolving 3-threshold access structure given by Theorem 2.14, and for $n \in \mathbb{N}$ let $\Pi_{\mathrm{LVW}}(n)$ denote the (finite) secret-sharing scheme for 2-slice access structures with n parties given by Theorem 2.9. We further set the generation sizes to be $g_1 = 16$ and $g_{i+1} = 2^{\log^c g_i}$ for all $i \geq 1$, where c is a constant defined in the analysis below, and G_i denote the set of parties in the i^{th} generation. Finally, let k be a sufficiently large constant to be determined by the analysis, and let $\tilde{g}_i = \sum_{j=i}^{i+k-1} g_j$ for all $i \in \mathbb{N}$.

Construction 5.5 (An Evolving Secret-Sharing Scheme $\Pi_{\text{2-slice}}$ for 2-Slice Access Structures)**.**
Input: A secret $s \in \{0,1\}$.
Sharing algorithm:

- Run Π_3, and let sh_t^3 denote the share of p_t.
- For every $i \in \mathbb{N}$, when the first party of generation G_i arrives, run $\Pi_{\text{LVW}}(\tilde{g}_i)$ for the parties in $G_i \cup \ldots \cup G_{i+k-1}$. For all t such that $p_t \in G_i \cup \ldots \cup G_{i+k-1}$, we denote its share by $\mathsf{sh}_{i,t}^{\text{LVW}}$. Note that each party p_t will receive k such shares, namely, one for each index $j \in \{i-k+1, \ldots, i\}$ such that $p_t \in G_i$.
- When party p_t arrives do:
  - Sample a uniform random bit r_t.
  - Let i be such that $p_t \in G_i$. If $i \geq 3$, then for every $t' < t$ such that $\{p_{t'}, p_t\} \in \Gamma$ and $p_{t'} \in G_j$ for some $1 \leq j \leq i-k$, set $\mathsf{sh}_{t',t}^{\text{pair}} = r_{t'} \oplus s$. Let $\mathsf{sh}_t^{\text{pair}}$ denote the vector of all such shares.
  - Define its share to be

$$\mathsf{sh}_t := \left(\mathsf{sh}_t^3,\ \left(\mathsf{sh}_{j,t}^{\text{LVW}}\right)_{j=i-k+1}^{i},\ r_t,\ \mathsf{sh}_t^{\text{pair}}\right),$$

    where $i \in \mathbb{N}$ is such that $p_t \in G_i$.

We next show that $\Pi_{\text{2-slice}}$ is an evolving secret-sharing scheme realizing Γ. We first prove the correctness of the scheme. Let $A \in \Gamma$ be a qualified set. If $|A| \geq 3$ then the parties can reconstruct the secret by the correctness of the 3-evolving sharing scheme Π_3. Otherwise, $A = \{p_{t'}, p_t\} \in \Gamma$ for some $t < t'$. If $t' \in G_i$ and $t \in G_i \cup \ldots \cup G_{i+k-1}$ for some $i \in \mathbb{N}$, then the parties can reconstruct s due to the correctness of $\Pi_{\text{LVW}}(\tilde{g}_i)$. Otherwise, the parties can reconstruct the secret since $p_{t'}$ holds $r_{t'}$ and p_t holds $r_{t'} \oplus s$.

Next, we prove the security of the scheme. Let $A \notin \Gamma$ be an unauthorized set. Then either $A = \{p_t\}$ or $A = \{p_{t'}, p_t\}$ for some $t' < t$. We may assume the latter without loss of generality. First, shares generated by Π_3 and Π_{LVW} give no information on the secret due to the security of the schemes Π_3 and Π_{LVW}. Moreover, the bits $r_j \oplus s$ given to $p_{t'}$ and p_t are all uniformly random and independent. Thus, the parties gain no information on s.

It is left to analyze the share complexity and choose the constant c, which governs the growth of the generations. Fix $t \in \mathbb{N}$ and let $i \in \mathbb{N}$ be such that $p_t \in G_i$. We assume that $i > k$. First, observe that since $\log g_{j+1} = \log^c g_j$ for all $j \in \mathbb{N}$, it follows that $\log g_{j+2} = \log^c g_j = (\log g_j)^{c^2}$, hence

$$\log g_{i+k-1} = (\log g_i)^{c^{k-1}} \tag{8}$$

and

$$\log g_{i-k} = (\log g_i)^{1/c^k}. \tag{9}$$

Moreover, since t is in the i^{th} generation, it follows that $t \geq g_{i-1}$. Therefore,

$$\log g_i = \log^c g_{i-1} \leq \log^c t. \tag{10}$$

Now, by Theorem 2.14 it holds that $|\mathsf{sh}^3_t| \leq 3\log t$ for all sufficiently large t's. By Theorem 2.9 it holds that

$$\left|\left(\mathsf{sh}^{\mathrm{LVW}}_{j,t}\right)^{i}_{j=i-k+1}\right| \leq 2^{O(\sqrt{\log g_{i+k-1}\cdot\log\log g_{i+k-1}})} = 2^{O\left(\sqrt{(\log t)^{c^k}\cdot\log\log t}\right)},$$

where the equality is by Equations (8) and (10). Finally, observe that

$$\mathsf{sh}^{\mathrm{pair}}_t = \sum_{j=1}^{i-k} g_j \leq \tilde{O}(g_{i-k}) = \tilde{O}\left(2^{(\log g_i)^{1/c^k}}\right) \leq \tilde{O}\left(2^{(\log t)^{1/c^{k-1}}}\right),$$

where the second equality is by Equation (9), and the last inequality is by Equation (10). Then the share complexity is optimized when

$$\tilde{O}\left(2^{(\log t)^{1/c^{k-1}}}\right) = 2^{O\left(\sqrt{(\log t)^{c^k}\cdot\log\log t}\right)},$$

which, up to polylogarithmic factors, holds if $(\log t)^{1/c^{k-1}} = (\log t)^{c^k/2}$, i.e., $c = 2^{1/(2k-1)}$. Therefore, the share size of p_t is at most

$$2^{O\left((\log t)^{2^{\frac{k}{2k-1}-1}}\cdot\sqrt{\log\log t}\right)} = 2^{O\left((\log t)^{2^{-\frac{k-1}{2k-1}}}\cdot\sqrt{\log\log t}\right)}.$$

Taking a sufficiently large k results in a share of size at most

$$2^{O\left((\log t)^{\frac{1}{\sqrt{2}}+\varepsilon}\cdot\sqrt{\log\log t}\right)}.$$

□

5.3 Scheme for Evolving 3-Slice Access Structures

In this section, we present our construction of an evolving secret-sharing scheme for the evolving 3-slice access structure. We prove the following.

Theorem 5.6. *Let Γ be an evolving 3-slice access structure. Then for every $\varepsilon > 0$, there exists an evolving secret-sharing scheme realizing Γ, such that for all $t \in \mathbb{N}$, the share size of p_t is at most $2^{O\left((\log t)^{\frac{1}{\sqrt{2}}+\varepsilon}\cdot\log\log t\right)}$.*

We first introduce in Section 5.3.1 a special kind of finite secret-sharing scheme, where the authorized and unauthorized sets do not necessarily cover all subsets of the parties. The proof of Theorem 5.6 is given below in Section 5.3.1.

5.3.1 A Special Secret-Sharing Scheme

Toward constructing an evolving secret-sharing scheme for 3-slice evolving access structures, we need a 3-server CDS protocol (viewed as a secret-sharing scheme) with additional security properties.[7] . In the following, for a set A and a number $1 \leq k \leq |A|$, we let $\binom{A}{k}$ denote the set of all subsets of A of size k.

[7] It is also possible to use a robust CDS [4], which guarantees security even if the referee sees multiple messages of the same server computed on different inputs with the same randomness. This, however, would result in an overall less efficient evolving scheme for evolving 3-slice access structures.

Lemma 5.7. *Let $n \in \mathbb{N}$, let $\mathcal{F} = \{\varphi : \binom{[n]}{2} \to \{0,1\}\}$, and let $P = \{p_1, \ldots, p_n\}$ and $P' = \{p'_\varphi\}_{\varphi \in \mathcal{F}}$ be two sets of parties. Define the 3-slice access structure Γ whose minimal authorized sets of size 3 are*

$$\left\{A \subseteq P \cup P' : \exists p_{t_1}, p_{t_2}, p'_\varphi \in A \text{ such that } \varphi(\{t_1, t_2\}) = 1\right\}.$$

Then there exists a secret-sharing scheme realizing Γ in which the share size of each party is at most $2^{O(\sqrt{\log n} \cdot \log\log n)}$. That is, the scheme satisfies the following.

1. *Any subset $A = \{p_{t_1}, p_{t_2}, p'_\varphi\}$ such that $\varphi(\{t_1, t_2\}) = 1$, the secret s can be reconstructed by the parties in A.*
2. *Any subset $A = \{p_{t_1}, p_{t_2}, p'_\varphi\}$ such that $\varphi(\{t_1, t_2\}) = 0$, learns nothing about s (i.e., the distribution of the parties' shares is independent of the secret s).*
3. *Any subset $A \subseteq P \cup P'$ such that $|A \cap P| < 2$ or $A \cap P' = \emptyset$, learns nothing about the secret.*

The construction is given below. We will use the following simple fact due to [12]. We prove it for completeness.

Fact 5.8. *For all sufficiently large $n \in \mathbb{N}$ there exists $\ell = \lfloor \log(n+1) \rfloor + 1$ partitions of $[n]$, denoted $\mathcal{K}_1 = \{K_{1,1}, \ldots, K_{1,k}\}, \ldots, \mathcal{K}_\ell = \{K_{\ell,1}, \ldots, K_{\ell,k}\}$, such that the following holds. For every $A \in \binom{[n]}{2}$ there exists $i \in [\ell]$ such that for every $j \in [2]$ it holds that $|A \cap K_{i,j}| = 1$.*

Proof. We define the i^{th} partition $\mathcal{K}_i$ according to the binary expansion of each number in $[n]$. Formally, for every $m \in [n]$ let $m_1, \ldots, m_\ell$ be its binary expansion. Then for $b \in \{0,1\}$ we define $K_{i,b} = \{m \in [n] : m_i = b\}$. Clearly, for every $A = \{m, m'\}$ where $m \neq m'$, it holds that for the bit $i \in [\ell]$ such that $m_i \neq m'_i$, that m and m' belong to different subsets of the partition $\mathcal{K}_i$. □

We next use Fact 5.8 to prove Lemma 5.7.

Proof of Lemma 5.7. We first explain the idea behind the constriction. First, share s in a 4-out-of-$(n + \binom{n}{2})$ secret-sharing scheme. Next, share it in 3-out-of-3 sharing scheme, and let sh'_1, sh'_2, and sh'_3 denote the shares. Now, consider a partitioning of the of the set of parties $P = K_1 \cup K_2$. The dealer runs a 3-server CDS protocol for the predicate INDEX^3_n with the secret sh'_1 (where each function $\varphi : \binom{[n]}{2} \to \{0,1\}$ is viewed as a two-dimensional array). The messages of the CDS protocol are then given to the parties as part of their shares depending on which server they correspond to, that is, let $\mathsf{cds}_{j,t}$ denote the message of server $j \in \{1,2\}$ on input $t \in [n]$, and let cds_φ denote the message of the last server on input φ (viewed as a database). Then p_t is given $\mathsf{cds}_{j,t}$ if $p_t \in K_j$, and p_φ is given cds_φ. This means that a set of parties $\{p_{t_1}, p_{t_2}, p'_\varphi\}$ can reconstruct sh'_1 only if $\varphi(\{t_1, t_2\}) = 1$ and t_1 and t_2 are in different sets of the partition. To ensure that reconstruction is possible if *any* pair arrives, we use the $\ell = O(\log n)$ partitions guaranteed to exist by Fact 5.8 and apply the above scheme ℓ times independently.

Next, the dealer then gives sh'_2 to all parties p'_φ from P', ensuring that nothing is learned about the secret without one of those parties. Finally, share sh'_3 in a 2-out-of-2 secret-sharing scheme, and consider one of the partitions of the set of parties P. Note that if the dealer gives each share to all parties in each of the sets in the partition, then reconstruction is possible only if at least one party from each set in the partition arrives. Similarly to before, this is done ℓ times independently.

We next present the formal construction. We first introduce some notations. Let $\mathcal{K}_1, \ldots, \mathcal{K}_\ell$ be the $\ell = O(\log n)$ partitions of $[n]$ guaranteed to exist by Fact 5.8. For $i \in [\ell]$ and $j \in \{1,2\}$ let

$K_{i,j} \in \mathcal{K}_i$ denote the j^{th} set of the i^{th} partition $\mathcal{K}_i$. Finally, let $\mathcal{P}_{\mathsf{CDS}}$ be the 3-server CDS protocol for INDEX_n^3 given by Theorem 5.3.

Construction 5.9 (A Secret-Sharing Scheme $\Pi_{\mathsf{sh\text{-}ind}}(n)$)**.**
Input: A secret $s \in \{0,1\}$.
The sharing algorithm:

- Share s in a 4-out-of-$(n + \binom{n}{2})$ secret-sharing scheme among $P \cup P'$. Let sh'_t denote the share of p_t for every $1 \le t \le n$, and let sh'_φ denote the share of p'_φ for every $\varphi \in \mathcal{F}$.
- For every $i \in [\ell]$ do:
  - Sample a pair of random bits r_i and r'_i uniformly at random, and let $s_i = s \oplus r_i \oplus r'_i$.
  - Share r_i in a 2-out-of-2 secret-sharing scheme to obtain shares $r_{i,1}$ and $r_{i,2}$.
  - Run CDS protocol $\mathcal{P}_{\mathsf{CDS}}$ for the function INDEX_n^3 and the secret s_i. For every $j \in \{1,2\}$ and $t \in [n]$ let $\mathsf{cds}_{i,j,t}$ denote the message sent by the j^{th} server on input t, and for every 2-dimensional database $D \in \{0,1\}^{n^2}$ let $\mathsf{cds}_{i,D}$ denote the message sent by the last server on input D.
- For each $t \in [n]$, set the share sh_t of p_t as follows. For every $i \in [\ell]$ let $j_{i,t} \in \{1,2\}$ be the unique index such that $t \in K_{i,j}$. Then
$$\mathsf{sh}_t = \left(\mathsf{sh}'_t, r_{i,j_{i,t}}, \mathsf{cds}_{i,j_{i,t},t}\right)_{i \in [\ell]}.$$
- For each $\varphi :\in \mathcal{F}$, set the share sh'_φ of p'_φ as follows. Let $D_\varphi \in \{0,1\}^{n^2}$ be the 2-dimensional database defined as
$$D_\varphi[t_1,t_2] = \begin{cases} \varphi(\{t_1,t_2\}) & \text{if } t_1,t_2 \text{ are distinct} \\ 0 & \text{otherwise} \end{cases}.$$
Then
$$\mathsf{sh}_\varphi = \left(\mathsf{sh}'_\varphi, r'_i, \mathsf{cds}_{i,D_\varphi}\right)_{i \in [\ell]}.$$

We next analyze the above scheme, showing that it realizes Γ. We first show correctness. By the use of the 4-out-of-4 secret-sharing scheme, we may consider authorized sets of size exactly 3, i.e., we prove Item 1. Fix $A = \{p_{t_1}, p_{t_2}, p'_\varphi\}$such that $\varphi(\{t_1,t_2\}) = 1$. By the way we chose the partitions $\mathcal{K}_1, \ldots, \mathcal{K}_\ell$, there exists $i \in [\ell]$ such that for every $j \in \{1,2\}$ it holds that $|A \cap P \cap K_{i,j}| = 1$. We next show that the parties in A can recover s_i, r_i, and r'_i, and thus can recover the secret s.

First, since each party in $A \cap P$ belongs to a different set in the partition $\mathcal{K}_i$, all shares of $(r_{i,j})_{j \in \{1,2\}}$ of r_i are held by A, thus r_i can be recovered. Second, since $p'_t \in A$, the parties hold r'_i as well. Finally, since $t_1 \neq t_2$ it holds that $D_\varphi[t_1,t_2] = \varphi(\{t_1,t_2\}) = 1$. Thus, by the correctness of the CDS protocol, the parties in A can reconstruct s_i.

We next prove the scheme is secure. We first show Item 2. Fix $A = \{p_{t_1}, p_{t_2}, p'_\varphi\}$ such that $\varphi(\{t_1,t_2\}) = 0$. We first show that for every $i \in [\ell]$, the shares of the parties that correspond to i cannot be used to recover the secret s. There are two cases to consider. In the first case, t_1 and t_2 belong to a different set in the partition $\mathcal{K}_i$. Here, since $D_\varphi[t_1,t_2] = \varphi(\{t_1,t_2\}) = 0$, by the security of the CDS protocol, the parties learn nothing about s_i, hence they cannot learn anything about s

using the i^{th} part of the share. In the second case, t_1 and t_2 belong to the same set in the partition $\mathcal{K}_i$ (and thus the CDS guarantee no security). Here, for the index $j \in \{1,2\}$ such that $t_1, t_2 \notin K_{i,j}$, no party is in $K_{i,j}$, thus no party obtains $r_{i,j}$. Therefore, the parties learn nothing about r_i, hence they cannot learn anything about s using the i^{th} part of the share. Finally, since for every i the i^{th} part of the shares are sampled independently, we conclude that the parties gain no information about the secret s.

We now prove Item 3. Here, simply observe that if $|A \cap P| < 2$, then for every $i \in [\ell]$ there exists $j \in \{1,2\}$ such that no party is in $K_{i,j}$, thus no party obtains $r_{i,j}$, and they learn nothing about r_i. Otherwise, if $A \cap P' = \emptyset$ then for every $i \in [\ell]$ no party in A receives r'_i. In both cases, the parties gain no information on s.

It is left to analyze the share complexity. For every $i \in [\ell]$, each party receives one bit and one message of the CDS protocol. Therefore, the share size of each party is at most

$$\ell \cdot 2^{O(\sqrt{\log n} \cdot \log\log n)} = 2^{O(\sqrt{\log n} \cdot \log\log n)}.$$

□

5.3.2 The Construction for Evolving 3-Slice Access Structures

In this section, we construct an evolving secret-sharing scheme for 3-slice access structures, thus proving Theorem 5.6.

Proof of Theorem 5.6. Let us first provide an intuitive description of the secret-sharing scheme. Similarly to the 2-slice evolving schemes, we first present a scheme whose complexity is worse than what is stated in Theorem 5.6. First, similarly to the 2-slice secret-sharing scheme, we handle authorized sets of size at least 4 using the scheme of Komargodski et al. [30]. For authorized sets of size exactly 3 we do the following. Partition the parties into generations and let G_i denote the i^{th} generation. Then, for every two consecutive generations, i.e., $G_i \cup G_{i+1}$ for $i \in \mathbb{N}$, we share the secret using a finite secret-sharing scheme for 3-slices.

It is left to handle authorized sets $\{p_{t_1}, p_{t_2}, p_{t_3}\}$ whose parties are not all in two adjacent generations. We separate this case into two more cases. We assume that $t_1 < t_2 < t_3$. In the first case, both p_{t_2} and p_{t_3} arrive at least 2 generation after p_{t_1}, i.e., $p_{t_1} \in G_{i_1}$, $p_{t_2} \in G_{i_2}$, and $p_{t_3} \in G_{i_3}$, where $i_1 + 1 < i_2 \leq i_3$. Here, we give p_{t_1} a random bit r_{t_1} and use the evolving 2-slice secret-sharing scheme from the previous section to share $s \oplus r_{t_1}$ among the parties that arrive from generation G_{i_1+2} onward.

Finally, we are left with the case where p_{t_1} and p_{t_2} arrive at two adjacent generations, and p_{t_3} arrives later, i.e., $p_{t_1}, p_{t_2} \in G_i \cup G_{i+1}$ and $p_{t_3} \in G_j$, where $j > i + 1$. Here, we use the sharing scheme given by Lemma 5.7. Specifically, we will use the sharing scheme to share the secret between $G_i \cup G_{i+1}$ and a set of parties $\{p'_\varphi\}_{\varphi \in \mathcal{F}}$, where $\mathcal{F} = \{\varphi : \binom{g_i + g_{i+1}}{2} \to \{0,1\}\}$. This results with shares for p_{t_1} and p_{t_2}, and with shares that are each associated with a function φ. We then associate with p_{t_3} the function that outputs 1 if and only if the two parties in the input complete p_{t_3} to an authorized set, and give to p_{t_3} all corresponding shares.

Now, similarly to the 2-slice secret-sharing scheme, we improve the above scheme by considering k adjacent generations instead of 2. We show that the scheme improves the larger k is.

We next present the formal construction. We first introduce some notations. Let Π_4 denote the evolving secret-sharing scheme for the evolving 4-threshold access structure given by Theorem 2.14, and for $n \in \mathbb{N}$ let $\Pi^{\mathsf{fin}}_{\text{3-slice}}(n)$ denote the (finite) secret-sharing scheme for 3-slice access structures

with n parties given by Theorem 2.10. We let $\Pi_{\text{2-slice}}$ denote the evolving secret-sharing scheme for evolving 2-slice access structures, in which the share size of p_t is at most $2^{O((\log t)^{\frac{1}{\sqrt{2}}+\varepsilon}\cdot\sqrt{\log\log t})}$ (such scheme exists by Theorem 5.4). We further set the generation sizes to satisfy $g_{i+1} = 2^{\log^c(g_i)}$ for all $i \geq 1$, where c is chosen in the analysis below, and G_i denote the set of parties in the i^{th} generation. Finally, let k be a sufficiently large constant to be determined by the analysis, and let $\tilde{g}_i = \sum_{j=i}^{i+k-1} g_i$ and $\mathcal{F}_i = \{\varphi : \binom{[\tilde{g}_i]}{2} \to \{0,1\}\}$ for all $i \in \mathbb{N}$.

Construction 5.10 (An Evolving Secret-Sharing Scheme $\Pi_{\text{3-slice}}$ for 3-Slice Access Structures)**.**
Input: The secret $s \in \{0,1\}$.
The sharing algorithm:

- Run Π_4, and let sh^4_t denote the share of p_t.
- For every $i \in \mathbb{N}$, when the first party of generation G_i arrives do the following.
  - Run $\Pi^{\mathsf{fin}}_{\text{3-slice}}(\tilde{g}_i)$ for the parties in $G_i \cup \ldots \cup G_{i+k-1}$. For all t such that $p_t \in G_i \cup \ldots \cup G_{i+k-1}$, we denote its share by $\mathsf{sh}^{\text{3-slice}}_{i,t}$. Note that each party p_t will receive k such shares, namely, one for each index $j \in \{i-k+1,\ldots,k\}$ such that $p_t \in G_i$.
  - Run $\Pi_{\mathsf{sh\text{-}ind}}(\tilde{g}_i, 2)$ for the parties $G_i \cup \ldots \cup G_{i+k-1} \cup \{p'_\varphi\}_{\varphi\in\mathcal{F}_i}$. For all t such that $p_t \in G_i \cup \ldots \cup G_{i+k-1}$, we denote its share by $\mathsf{sh}^{\mathsf{sh\text{-}ind}}_{i,t}$.[8] For every p'_φ we denote its share by $\mathsf{sh}^{\mathsf{sh\text{-}ind}}_{i,\varphi}$. For every $t' \in \mathbb{N}$ such that $p_{t'} \in G_j$ for some $j \geq i+k$, let $\varphi_{i,t'} : \binom{[\tilde{g}_i]}{2} \to \{0,1\}$ be defined as $\varphi_{i,t'}(\{t_1,t_2\}) = 1$ if and only if $\{p_{t_1}, p_{t_2}, p_{t'}\} \in \Gamma$.
- When party p_t arrives do:
  - Let $i \in \mathbb{N}$ be the index such that $p_t \in G_i$.
  - Sample a random bit r_t uniformly at random.
  - Run the evolving secret-sharing scheme $\Pi_{\text{2-slice}}$ to share the secret $r_t \oplus s$ among the parties $\bigcup_{j\geq i+k} G_i$, for the 2-slice evolving access structure whose minimal authorized sets of size 2 are $\{\{p_{t_1}, p_{t_2}\} : \{p_t, p_{t_1}, p_{t_2}\} \in \Gamma\}$. For every t' such that $p_{t'} \in G_j$ for $j \geq i+k$, denote its share by $\mathsf{sh}^{\text{2-slice}}_{t,t'}$.
  - Define its share to be

$$\mathsf{sh}_t = \left(\mathsf{sh}^4_t,\ \mathsf{sh}^{\text{3-slice}}_t,\ r_t,\ \mathsf{sh}^{\text{2-slice}}_t,\ \mathsf{sh}^{\mathsf{sh\text{-}ind}}_{i-1,t},\ \mathsf{sh}^{\mathsf{sh\text{-}ind}}_{i,t},\ \mathsf{sh}^{\mathsf{sh\text{-}ind}}_t\right),$$

    where $\mathsf{sh}^{\text{3-slice}}_t = (\mathsf{sh}^{\text{3-slice}}_{j,t})_{j\in[i-k]}$, where $\mathsf{sh}^{\text{2-slice}}_t = (\mathsf{sh}^{\text{2-slice}}_{t',t})_{t'\in G_j, j\in[i-k]}$, and where $\mathsf{sh}^{\mathsf{sh\text{-}ind}}_t = (\mathsf{sh}^{\mathsf{sh\text{-}ind}}_{j,\varphi_{j,t}})_{j\in[i-k]}$.

We next analyze the scheme. We first prove its correctness. Let $A \in \Gamma$ be an authorized set. First, if $|A| \geq 4$ then the parties can reconstruct s by the correctness of Π_4. We may now assume that $A = \{p_{t_1}, p_{t_2}, p_{t_3}\}$ where $t_1 < t_2 < t_3$. Let $i_1, i_2, i_3 \in \mathbb{N}$ be such that $p_{t_j} \in G_{i_j}$ for all $j \in [3]$. Note that $i_1 \leq i_2 \leq i_3$. We separate the proof into 3 cases.

Case 1: $i_2, i_3 \geq i_1 + k$**.** By the correctness of $\Pi_{\text{2-slice}}$ it holds that p_{t_2} and p_{t_3} can reconstruct $r_{t_1} \oplus s$. Since p_{t_1} receives r_{t_1}, they can reconstruct s.

[8] Similarly to before, p_t will receive k such shares.

Case 2: $i_3 \leq i_1 + k - 1$**.** In this case, all three parties belong to k consecutive generations. Therefore, they reconstruct s due to the correctness of $\Pi^{\mathsf{fin}}_{\text{3-slice}}$.

Case 3: $i_2 \leq i_1 + k - 1$ **and** $i_3 \geq i_1 + k$**.** Observe that p_{t_3} holds $\mathsf{sh}^{\mathsf{sh\text{-}ind}}_{i_1,\varphi_{i_1,t_3},t_3}$, where φ_{i_1,t_3} satisfies $\varphi_{i_1,t_3}(\{t_1,t_2\}) = 1$. Since p_{t_1} and p_{t_2} hold $\mathsf{sh}^{\mathsf{sh\text{-}ind}}_{i_1,t_1}$ and $\mathsf{sh}^{\mathsf{sh\text{-}ind}}_{i_1,t_2}$, respectively, it follows that the parties can reconstruct s due to the correctness of $\Pi_{\mathsf{sh\text{-}ind}}$.

Next, we prove the security of the scheme. Let $A \notin \Gamma$ be an unauthorized set. We may assume without loss of generality that $A = \{p_{t_1}, p_{t_2}, p_{t_3}\}$, where $t_1 < t_2 < t_3$. Let $i_1, i_2, i_3 \in \mathbb{N}$ be such that $p_{t_j} \in G_{i_j}$ for all $j \in [3]$, and note that $i_1 \leq i_2 \leq i_3$. First, notice that the shares generated by Π_4 and $\Pi^{\mathsf{fin}}_{\text{3-slice}}$ give no information on the secret, due to the security of these schemes. Since the other shares are generated independently, we may now analyze only them. We next separate the proof into two cases. We note that in both cases, all other shares that are not discussed are independent and reveal no information on s.

Case 1: $i_2, i_3 \geq i_1 + k$**.** Since $A \notin \Gamma$, it follows by the security of $\Pi_{\text{2-slice}}$ that p_{t_2} and p_{t_3} learn nothing about $r_{t_1} \oplus s$.

Case 2: $i_3 \leq i_1 + k - 1$**.** In this case, all parties belong to k consecutive generations. Then by Item 3 of Lemma 5.7 the shares generated by $\Pi_{\mathsf{sh\text{-}ind}}$ reveal no information to the parties.

Case 3: $i_2 \leq i_1 + k - 1$ **and** $i_3 \geq i_1 + k$**.** In this case, p_{t_3} obtains $\mathsf{sh}^{\mathsf{sh\text{-}ind}}_{i_1,\varphi_{i_1,t_3},t_3}$. Since $\varphi_{i_1,t_3}(\{t_1,t_2\}) = 0$, by Item 2, the shares generated by $\Pi_{\mathsf{sh\text{-}ind}}$ reveal no information on the secret.

It is left to analyze the share complexity. Fix $t \in \mathbb{N}$ and let $i \in \mathbb{N}$ be such that $p_t \in G_i$. We assume that $i > 2$. First, similarly to the analysis of our evolving secret-sharing scheme for evolving 2-slice access structures, First, observe that since $\log g_{j+1} = \log^c g_j$ for all $j \in \mathbb{N}$, it follows that

$$\log g_{i+k-1} = (\log g_i)^{c^{k-1}} \tag{11}$$

and that

$$\log g_{i-k} = (\log g_i)^{1/c^k}. \tag{12}$$

Moreover, since $t \geq g_{i-1}$, it further holds that

$$\log g_i = \log^c g_{i-1} \leq \log^c t. \tag{13}$$

Now, by Theorem 2.14 it holds that $|\mathsf{sh}^4_t| \leq 4 \log t$ for all sufficiently large t. By Theorem 2.10 it holds that

$$|\mathsf{sh}^{\text{3-slice}}_t| \leq 2^{O(\sqrt{\log g_{i+k-1}} \cdot \log\log g_{i+k-1})} \leq 2^{O\left((\log t)^{c^k/2} \cdot \log\log t\right)}, \tag{14}$$

where the last inequality is by Equations (11) and (13). Next, by Lemma 5.7 it holds that

$$\begin{aligned} |\mathsf{sh}^{\mathsf{sh\text{-}ind}}_{i-1,t}| + |\mathsf{sh}^{\mathsf{sh\text{-}ind}}_{i,t}| + |\mathsf{sh}^{\mathsf{sh\text{-}ind}}_{t}| &\leq 2^{O(\sqrt{\log g_{i+k-1}} \cdot \log\log g_{i+1})} \\ &\leq 2^{O\left((\log t)^{c^k/2} \cdot \log\log t\right)}, \end{aligned} \tag{15}$$

where the last inequality is by Equations (11) and (13). Finally, by Theorem 5.4 it holds that

$$|\mathsf{sh}_t^{2\text{-slice}}| \leq \tilde{O}(g_{i-k}) \cdot 2^{O\left((\log t)^{\frac{1}{\sqrt{2}}+\varepsilon} \cdot \sqrt{\log\log t}\right)} \leq 2^{O\left((\log t)^{1/c^{k-1}} + (\log t)^{\frac{1}{\sqrt{2}}+\varepsilon} \cdot \sqrt{\log\log t}\right)}, \tag{16}$$

where the last inequality is by Equations (12) and (13).

The share size is optimized if all expressions in Equations (14) to (16) are equal. Interestingly, equality occurs when the parameters c and k are chosen as in our construction for the evolving 2-slice access structures. Indeed, observe that for $c = 2^{1/(2k-1)}$ and a sufficiently large k, it holds that

$$(\log t)^{c^k/2} = (\log t)^{1/c^{k-1}} \leq (\log t)^{\frac{1}{\sqrt{2}}+\varepsilon}.$$

Therefore, up to polylogarithmic factors, the share size is optimized for $c = 2^{1/(2k-1)}$. Thus, the share size of p_t is at most

$$2^{O\left((\log t)^{\frac{1}{\sqrt{2}}+\varepsilon} \cdot \log\log t\right)}.$$

□

6 Lower Bounds for Evolving Secret Sharing

In this section, we prove lower bounds on the share size for explicit evolving access structures. Toward proving these results, we first show a general lower bound. This lower bound generalizes the recent result of [35] to include more access structures, and is inspired by the generalization of Csirmaz's lower bound [20] due to Blundo et al. [16]. In Section 6.1 we present the general lower bound. Then, in Section 6.2 we use our general result to prove lower bound for evolving directed st-connectivity and for k-hypergraph (for a constant k). Interestingly, we also obtain a lower bound of $\Omega(\sqrt{n})$ over the maximal share complexity for finite (i.e., not evolving) directed st-connectivity. Previously, this was known only for linear secret-sharing schemes [11].

6.1 Lower Bounds for General Access Structures

In this section, we present a general lower bound for evolving secret-sharing schemes. We first define the notion of an independent sequence.

Definition 6.1 (Independent Sequences [16])**.** *Let $n, \ell \in \mathbb{N}$ be integers, let $A \subseteq \{p_{\ell+1}, \ldots, p_n\}$ be a set of parties, and let Γ be an access structure whose set of parties is $\{p_1, \ldots, p_n\}$. An* independent sequence of length ℓ with respect to A *is a sequence $A_1, \ldots, A_\ell \subseteq A$ of subsets of A such that the following hold.*

1. $\{p_1, \ldots, p_i\} \cup A_i \in \Gamma$ *for all* $i \in \{1, \ldots, \ell\}$.
2. $\{p_1, \ldots, p_{i-1}\} \cup A_i \notin \Gamma$ *for all* $i \in \{1, \ldots, \ell\}$.

Remark 6.2. Blundo et al. [16] originally called the set of parties $\{p_1, \ldots, p_\ell\}$ an independent set. We decided to refer to the sequence of subsets $A_1, \ldots, A_\ell$ as we find it more informative.

With the definition of independent sequence in mind, Blundo et al.[16] showed a lower bound for any access structure containing a "long" independent sequence, generalizing the lower bound of Csirmaz [20]. Formally, they proved the following.

Theorem 6.3 ([16, 20])**.** *Let Γ be an access structure whose set of parties is $P = \{p_1, \ldots, p_n\}$. Assume there exists $\ell \in [n]$ and $A \subseteq \{p_{\ell+1}, \ldots, p_n\}$, for which there exists an independent sequence of length ℓ with respect to A. Then for every secret-sharing scheme realizing Γ, the total share size of the parties in A is at least $\ell - 1$.*

In order to show a lower bound for evolving access structures, we first extend the notion of independent sequence to a an infinite independent sequence.

Definition 6.4 (Infinite Independent Sequences)**.** *Let Γ be an evolving access structure whose set of parties is $P = \{p_i : i \in \mathbb{N}\}$, let $A = \{p_{a_1}, p_{a_2}, \ldots\}$ and $B = \{p_{b_1}, p_{b_2}, \ldots\}$ be disjoint subsets of P, where $a_i < a_{i+1}$ and $b_i < b_{i+1}$ for all $i \in \mathbb{N}$. For a function $\ell : \mathbb{N} \to \mathbb{N}$, an $\ell(\cdot)$-infinite independent sequence with respect to A is an infinite sequence $\mathcal{A} = (A_i)_{i \in \mathbb{N}}$, where $A_i \subseteq A$ for all $i \in \mathbb{N}$, such that the following hold.*

1. *For every $i \in \mathbb{N}$ there are exactly $\ell(i)$ sets from $\mathcal{A}$ that contain only the first i parties from A, i.e., from $\{p_{a_1}, \ldots, p_{a_i}\}$.*
2. *$\{p_{b_1}, \ldots, p_{b_i}\} \cup A_i \in \Gamma$ for all $i \in \mathbb{N}$.*
3. *$\{p_{b_1}, \ldots, p_{b_{i-1}}\} \cup A_i \notin \Gamma$ for all $i \in \mathbb{N}$.*

We are now ready to prove our result. Roughly, we show that if an evolving access structure admits an $\ell(\cdot)$-infinite independent sequence for a fast-increasing function $\ell(\cdot)$, then we obtain a large lower bound on the share size of the parties.

Theorem 6.5. *Let Γ be an evolving access structure over a set of parties $P = \{p_t\}_{t \in \mathbb{N}}$, let $A = \{p_{a_1}, p_{a_2}, \ldots\}$ and $B = \{p_{b_1}, p_{b_2}, \ldots\}$ be disjoint subsets of P, where $a_i < a_{i+1}$ and $b_i < b_{i+1}$ for all $i \in \mathbb{N}$. Assume that the following hold*

1. *The order of arrival in Γ is as follows. Let $a_1 = 1$, $b_1 = 3$, for every $i \in \mathbb{N}$ let a_i is the smallest n such that $a_j \neq n$ and $b_j \neq n$ for all $j < i$, and $b_i = a_i + 2^i$.*[9]
2. *There exists a function $\ell : \mathbb{N} \to \mathbb{N}$, for which there exists an $\ell(\cdot)$-infinite independent sequence with respect to A.*

Then for every evolving secret-sharing scheme realizing Γ, it holds that for every $t \in \mathbb{N}$, the total share size of the first t parties is at least $\ell(t - \log t) + \log t - 1$. In particular, there exists a party p_j, where $j \leq t$, whose share size is at least $\ell(t - \log t)/t + o(1)$.

Proof. Fix $t \in \mathbb{N}$, and let i denote the largest number such that $a_i \leq t$. Stated differently, i is the number of parties from A that arrive by time t. Consider the finite access structure Γ^{b_i} that is induced by the first b_i parties of Γ. Then the sequence of $\ell(i)$ sets that contain only parties from $\{p_{a_1}, \ldots, p_{a_i}\}$ forms an independent set of length $\ell(i)$ with respect to $\{p_{a_1}, \ldots, p_{a_i}\}$. Therefore, by Theorem 6.3 the total share size of the parties in $\{p_{a_1}, \ldots, p_{a_i}\}$ is at least $\ell(i) - 1$.

Now, for the evolving scheme, the total share size of the first t parties must be at least $\ell(i) - 1 + t - i$, since there are exactly $t - i$ parties from B that arrive by time t. It is left to approximate i using t. We next show that a_j is roughly $j + \log j$ for every j. First, observe that for every j it holds that

$$a_j - j = |\{b \in \mathbb{N} : p_b \in B \wedge b < a_j\}|$$

[9] The choice of 2^i is arbitrary, and choosing any sufficiently fast increasing function instead of 2^i would suffice.

is the number of parties from B that arrive before p_{a_j}. Second, by a simple inductive argument, it holds that $j \leq a_j < 2j$ for all $j \in \mathbb{N}$. Therefore, for every j is holds that

$$b_{\log(j/2)} = \frac{j}{2} + a_{\log(j/2)} < j < a_j < 2j < 2j + a_{\log(2j)} = b_{\log(2j)}.$$

Since for every k, it further holds that the number of parties from B that arrive until time b_k is exactly k, it follows that number of parties from B that arrive until time a_j is between $\log(j/2)$ and $\log(2j)$. Therefore

$$j + \log j - 1 \leq a_j \leq j + \log j + 1,$$

for every j.

Now, by the maximality of i, it follows that $a_i \leq t \leq a_{i+1} - 1$. Therefore

$$i + \log i - 1 \leq t \leq i + 1 + \log(i+1).$$

This implies that $t - \log t \leq i \leq t + \log t$. Since $\ell(\cdot)$ is non-decreasing, we conclude that the total share of size the first t parties is at least

$$\ell(i) - 1 + t - i \geq \ell(t - \log t) + \log t - 1.$$

□

6.2 Lower Bounds for Directed st-Connectivity and k-Hypergraphs

In this section, we use Theorem 6.5 to obtain lower bounds on the share sizes of directed st-connectivity and k-hypergraphs access structures (for a constant k). As stated earlier, we also show how to use Theorem 6.3 to obtain a lower bound for finite directed st-connectivity. We first present this lower bound.

Theorem 6.6 (Lower Bounds for Directed st-connectivity)**.** *For every $n \in \mathbb{N}$ there exists a directed st-connectivity access structure Γ with n parties, such that for every secret-sharing scheme realizing Γ, there exists at least one party whose share size is at least $\frac{n-1}{2\sqrt{n}}$.*

Proof. Let $h = \lceil \sqrt{n} \rceil$ and consider the graph $G = (V, E)$, where

$$V = \{u_s, u_t\} \cup \{u_1, \ldots, u_h\} \cup \{v_1, \ldots, v_h\},$$

and where

$$E = \{(u_s, u_i) : 1 \leq i \leq h\} \cup \{(u_i, v_j) : 1 \leq i, j \leq h\} \cup \{(v_j, u_t) : 1 \leq j \leq h\}.$$

See Figure 3 for an illustration of the graph. We let Γ be the st-connectivity access structure associated with the graph G, namely, the parties are all edges, and a set is authorized if and only if it contains a path from u_s to u_t. Let A be the set of all edges where one of their endpoints includes u_s or u_t, i.e.,

$$A = \{(u_s, u_i) : 1 \leq i \leq h\} \cup \{(v_j, u_t) : 1 \leq j \leq h\}.$$

We next show that there exists an independent sequence of length h^2 with respect to A. For every $i, j \in [h]$, let $A_{i,j} = \{(u_s, u_i), (v_j, u_t)\}$. Then $\{(u_i, v_j)\} \cup A_{i,j} \in \Gamma$. Additionally, for every set

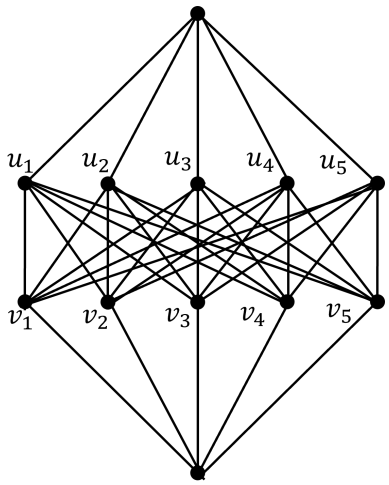

Figure 3: An illustration of the graph with the independent sequence.

of edges B that does not contain (u_i, v_j), it holds that $B \cup A_{i,j} \notin \Gamma$. We conclude that $(A_{i,j})_{i,j\in[h]}$ is an independent sequence of length h^2 with respect to A. Therefore, by Theorem 6.3, for every secret-sharing scheme realizing Γ the total share size of the parties in A is at least $h^2 - 1$, hence there exists a party whose share size is at least

$$\frac{h^2 - 1}{|A|} \geq \frac{n-1}{2\sqrt{n}}.$$

□

We note that using the techniques of [20], one can prove a lower bound of $\Omega(n^{3/2})$ on the total share size of st-connectivity.

We next show a lower bound for the evolving case.

Theorem 6.7 (Lower Bounds for Evolving Directed st-connectivity)**.** *There exists an evolving directed st-connectivity access structure Γ, such that for every evolving secret-sharing scheme realizing Γ, for every sufficiently large t it holds that the total share size of the first t party is at least $\Omega(t^2)$.*

Proof. Consider the graph $G = (V, E)$, where

$$V = \{u_s, u_t\} \cup \{u_i : i \in \mathbb{N}\} \cup \{v_j : j \in \mathbb{N}\},$$

and where

$$E = \{(u_s, u_i) : i \in \mathbb{N}\} \cup \{(u_i, v_j) : i, j \in \mathbb{N}\} \cup \{(v_j, u_t) : j \in \mathbb{N}\}.$$

Let $S = \{(u_s, u_i) : i \in \mathbb{N}\}$, let $T = \{(v_j, u_t) : j \in \mathbb{N}\}$, and let $B = E \setminus (S \cup T)$. We first define an ordering on $S \cup T$ and B. The order inside $S \cup T$ is such that its i^{th} party is $(u_s, u_{(i-1)/2})$ if i is odd, and is $(v_{i/2}, u_t)$, otherwise. Let $f : \mathbb{N} \to \mathbb{N}^2$ be a bijection. The order inside B is such that the i^{th} party is (u_{i_1}, v_{i_2}), where $f(i) = (i_1, i_2)$. We let Γ be the evolving directed st-connectivity access structure associated with the graph G, where the order of arrival is defined to satisfy Item 1 from Theorem 6.5.

We next show that there exists an infinite independent sequence $\mathcal{A} = (A_i)_{i\in\mathbb{N}}$ with respect to $S\cup T$. For every $i \in \mathbb{N}$, let $A_i = \{(u_s, u_{i_1}), (v_{i_2}, u_t)\}$, where $(i_1, i_2) = f(i)$. Then $\{(u_{i_1}, v_{i_2})\}\cup A_i \in \Gamma$. Additionally, for every set of edges $B' \subseteq B \setminus \{(u_{i_1}, v_{i_2})\}$, it holds that $B' \cup A_i \notin \Gamma$. Next, let $\ell(i)$ denote the number of sets from the sequence $\mathcal{A}$ that contain only the first i parties from $S \cup T$. Observe that $\ell(i) \geq \Omega(i^2)$. Therefore, $\mathcal{A}$ is an $\ell(\cdot)$-infinite independent sequence. By Theorem 6.5, it follows that there exists a reordering of the arrival time of the parties, such that for the resulting evolving access structure the total share size of the first t parties is at least

$$\ell(t - \log t) + \log t - 1 \geq \Omega(t - \log t)^2.$$

□

Finally, we show a lower bound for k-hypergraph access structure. This generalizes the result of Beimel [6] to the evolving case. Interestingly, the bound is optimal up to a factor of t (see Appendix A for a construction).

Theorem 6.8 (Lower Bounds for Evolving k-hypergraph access structure)**.** *For every $k \in \mathbb{N}$ there exists an evolving k-hypergraph access structure Γ, such that for every evolving secret-sharing scheme realizing Γ, it holds that for all sufficiently large $t \in \mathbb{N}$, the total share size of the first t parties in Γ is at least $\Omega\Big(\big(\frac{t}{k}\big)^{k-1}\Big)$.*

Proof. Consider the k-partite hypergraph $H = (V_1 \cup \ldots \cup V_{k-1} \cup B, E)$, where for every $j \in [k-1]$ we let

$$V_j = \{v_{j,i} : i \in \mathbb{N}\},$$

we let

$$B = \big\{v_{i_1,\ldots,i_{k-1}} : i_1, \ldots, i_{k-1} \in \mathbb{N}\big\},$$

and we let

$$E = \big\{\big(v_{1,i_1}, \ldots, v_{k-1,i_{k-1}}, v_{i_1,\ldots,i_{k-1}}\big) : i_1, \ldots, i_{k-1} \in \mathbb{N}\big\}.$$

Define the function $r : \mathbb{N} \to \mathbb{N}$ as $r(i) = i \bmod k$. We first define an ordering over A and B. The order inside A is such that its i^{th} party is $v_{r(i),(i-r(i))/k}$. Let $f : \mathbb{N} \to \mathbb{N}^k$ be a bijection. The order inside B is such that the i^{th} party is $v_{f(i)}$. We let Γ be the evolving k-hypergraph access structure associated with H, where the order of arrival is defined to satisfy Item 1 from Theorem 6.5.

Let $A = V_1 \cup \ldots \cup V_{k-1}$. We next show that there exists an infinite independent sequence $\mathcal{A} = (A_i)_{i\in\mathbb{N}}$ with respect to A. For every i we let $A_i = \{v_{1,i_1}, \ldots, v_{k-1,i_{k-1}}\}$, where $(i_1, \ldots, i_k) = f(i)$. Observe that $A_i \cup \{v_{f(i)}\} \in \Gamma$, and that for any $B' \subseteq B \setminus \{v_{f(i)}\}$ it holds that $A_i \cup B' \notin \Gamma$. Next, let $\ell(i)$ denote the number of sets from the sequence $\mathcal{A}$ that contain only the first i parties from A. Then $\ell(i) \geq \Omega((i/k)^{k-1})$. Therefore, $\mathcal{A}$ is an $\ell(\cdot)$-infinite independent sequence. By Theorem 6.5, there exists a reordering of the arrival time of the parties, such that for the resulting access structure, it holds that for every t the total share size of the first t parties is at least

$$\ell(t - \log t) + \log t - 1 \geq \Omega\left(\left(\frac{t}{k}\right)^{k-1}\right)$$

□

Bibliography

[1] Benny Applebaum and Barak Arkis. On the power of amortization in secret sharing: d-uniform secret sharing and CDS with constant information rate. In *TCC 2018*, volume 11239 of *LNCS*, pages 317–344, 2018.

[2] Benny Applebaum, Amos Beimel, Oriol Farràs, Oded Nir, and Naty Peter. Secret-sharing schemes for general and uniform access structures. In *EUROCRYPT 2019*, volume 11478 of *LNCS*, pages 441–471, 2019.

[3] Benny Applebaum, Amos Beimel, Oded Nir, and Naty Peter. Better secret sharing via robust conditional disclosure of secrets. In *STOC 2020*, pages 280–293, 2020.

[4] Benny Applebaum, Amos Beimel, Oded Nir, and Naty Peter. Better secret sharing via robust conditional disclosure of secrets. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, page 280–293, New York, NY, USA, 2020. Association for Computing Machinery.

[5] Benny Applebaum and Oded Nir. Upslices, downslices, and secret-sharing with complexity of 1.5^{n}. In *CRYPTO 2021*, volume 12827, pages 627–655. Springer, 2021.

[6] Amos Beimel. Lower bounds for secret-sharing schemes for k-hypergraphs. In Kai-Min Chung, editor, *ITC 2023*, volume 267 of *LIPIcs*, pages 16:1–16:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[7] Amos Beimel, Yuval Ishai, Ranjit Kumaresan, and Eyal Kushilevitz. On the cryptographic complexity of the worst functions. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 317–342. Springer-Verlag, 2014.

[8] Amos Beimel, Eyal Kushilevitz, and Pnina Nissim. The complexity of multiparty PSM protocols and related models. In *EUROCRYPT 2018*, volume 10821 of *LNCS*, pages 287–318, 2018.

[9] Amos Beimel and Hussien Othman. Evolving ramp secret-sharing schemes. In *SCN 2018*, volume 11035 of *LNCS*, pages 313–332. Springer-Verlag, 2018.

[10] Amos Beimel and Hussien Othman. Evolving ramp secret sharing with a small gap. In *EUROCRYPT 2020*, volume 12105 of *LNCS*, pages 529–555, 2020.

[11] Amos Beimel and Anat Paskin. On linear secret sharing for connectivity in directed graphs. In R. Ostrovsky, R. De Prisco, and I. Visconti, editors, *Proc. of the Sixth Conference on Security and Cryptography for Networks*, volume 5229 of *LNCS*, pages 172–184. Springer-Verlag, 2008.

[12] Amos Beimel and Naty Peter. Optimal linear multiparty conditional disclosure of secrets protocols. In *ASIACRYPT 2018*, volume 11274 of *LNCS*, pages 332–362, 2018.

[13] Josh Benaloh and Steven Rudich. Private communication, 1989.

[14] Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In *CRYPTO '88*, volume 403 of *LNCS*, pages 27–35, 1988.

[15] George Rober Blakley. Safeguarding cryptographic keys. *1979 International Workshop on Managing Requirements Knowledge (MARK)*, pages 313–318, 1979.

[16] Carlo Blundo, Alfredo De Santis, Roberto De Simone, and Ugo Vaccaro. Tight bounds on the information rate of secret sharing schemes. *Des. Codes Cryptography*, 11(2):107–122, 1997.

[17] Christian Cachin. On-line secret sharing. In *Proc. of the 5th IMA International Conference on Cryptography and Coding*, volume 1025 of *LNCS*, pages 190–198, 1995.

[18] Shion Samadder Chaudhury, Sabyasachi Dutta, and Kouichi Sakurai. Ac^0 constructions of secret sharing schemes - accommodating new parties. In *NSS 2020*, volume 12570 of *LNCS*, pages 292–308, 2020.

[19] László Csirmaz. The dealer's random bits in perfect secret sharing schemes. *Studia Sci. Math. Hungar.*, 32(3–4):429–437, 1996.

[20] László Csirmaz. The size of a share must be large. *J. Cryptol.*, 10(4):223–231, sep 1997.

[21] László Csirmaz and Gábor Tardos. On-line secret sharing. *Des. Codes Cryptography*, 63(1):127–147, 2012.

[22] Paolo D'Arco, Roberto De Prisco, and Alfredo De Santis. Secret sharing schemes for infinite sets of participants: A new design technique. *Theor. Comput. Sci.*, 859:149–161, 2021.

[23] Paolo D'Arco, Roberto De Prisco, Alfredo De Santis, Angel Pérez del Pozo, and Ugo Vaccaro. Probabilistic Secret Sharing. In *MFCS 2018*, volume 117 of *LIPIcs*, pages 64:1–64:16, 2018.

[24] Yvo Desmedt, Sabyasachi Dutta, and Kirill Morozov. Evolving perfect hash families: A combinatorial viewpoint of evolving secret sharing. In *Cryptology and Network Security*, pages 291–307, 2019.

[25] Sabyasachi Dutta, Partha Sarathi Roy, Kazuhide Fukushima, Shinsaku Kiyomoto, and Kouichi Sakurai. Secret sharing on evolving multi-level access structure. In *Information Security Applications*, pages 180–191, 2020.

[26] Paul Erdös and László Pyber. Covering a graph by complete bipartite graphs. *Discrete Mathematics*, 170(1–3):249–251, 1997.

[27] Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting data privacy in private information retrieval schemes. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 151–160, New York, NY, USA, 1998. Association for Computing Machinery.

[28] Mitsuru Ito, Akira Saito, and Takao Nishizeki. Secret sharing scheme realizing general access structure. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, 72(9):56–64, 1989.

[29] Mauricio Karchmer and Avi Wigderson. On span programs. In *8th Structure in Complexity Theory*, pages 102–111, 1993.

[30] Ilan Komargodski, Moni Naor, and Eylon Yogev. How to share a secret, infinitely. *IEEE Trans. Inf. Theory*, 64(6):4179–4190, 2018.

[31] Ilan Komargodski and Anat Paskin-Cherniavsky. Evolving secret sharing: Dynamic thresholds and robustness. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography*, pages 379–393, Cham, 2017. Springer International Publishing.

[32] Tianren Liu and Vinod Vaikuntanathan. Breaking the circuit-size barrier in secret sharing. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, page 699–708, New York, NY, USA, 2018. Association for Computing Machinery.

[33] Tianren Liu, Vinod Vaikuntanathan, and Hoeteck Wee. Conditional disclosure of secrets via non-linear reconstruction. In *CRYPTO 2017*, volume 10401 of *LNCS*, pages 758–790, 2017.

[34] Tianren Liu, Vinod Vaikuntanathan, and Hoeteck Wee. Towards breaking the exponential barrier for general secret sharing. In *EUROCRYPT 2018*, volume 10820 of *LNCS*, pages 567–596, 2018.

[35] Noam Mazor. A lower bound on the share size in evolving secret sharing. In Kai-Min Chung, editor, *4th Conference on Information-Theoretic Cryptography, ITC 2023, June 6-8, 2023, Aarhus University, Aarhus, Denmark*, volume 267 of *LIPIcs*, pages 2:1–2:9. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[36] Ryo Okamura and Hiroki Koga. New constructions of an evolving 2-threshold scheme based on binary or d-ary prefix codes. In *2020 International Symposium on Information Theory and Its Applications (ISITA)*, pages 432–436, 2020.

[37] Kittiphop Phalakarn, Vorapong Suppakitpaisarn, Nuttapong Attrapadung, and Kanta Matsuura. Evolving homomorphic secret sharing for hierarchical access structures. In *Advances in Information and Computer Security: 16th International Workshop on Security, IWSEC 2021, Virtual Event, September 8–10, 2021, Proceedings*, page 77–96, Berlin, Heidelberg, 2021. Springer-Verlag.

[38] Toniann Pitassi and Robert Robere. Lifting nullstellensatz to monotone span programs over any field. In *50th STOC*, pages 1207–1219, 2018.

[39] Adi Shamir. How to share a secret. In *Communications of the ACM, 22*, pages 612–613, 1979.

[40] Hung-Min Sun and Shiuh-Pyng Shieh. Secret sharing in graph-based prohibited structures. In *INFOCOM '97*, pages 718–724. IEEE, 1997.

[41] Chaoping Xing and Chen Yuan. Evolving secret sharing schemes based on polynomial evaluations and algebraic geometry codes. *IACR Cryptol. ePrint Arch.*, page 1115, 2021.

[42] Wei Yan, Sian-Jheng Lin, and Yunghsiang S. Han. A new metric and the construction for evolving 2-threshold secret sharing schemes based on prefix coding of integers. *IEEE Transactions on Communications*, 71(5):2906–2915, 2023.

A Evolving Secret-Sharing Scheme for Evolving k-Hypergraphs access structure

In this section, we complement our lower bound for the evolving k-hypergraphs by showing a simple evolving secret-sharing scheme with $O(t^{k-1})$ share size.

Theorem A.1. *Fix $k \in \mathbb{N}$. Then for every k-hypergraph evolving access structure Γ there exists an evolving secret-sharing scheme realizing Γ, such that for every $t \in \mathbb{N}$ the share size of the t^{th} party is $O(t^{k-1})$.*

Proof. Fix a k-hypergraph evolving access structure Γ. We share a secret $s \in \{0,1\}$ using the following simple scheme. For every $\ell \leq k-1$ and every sequence $j_1 < j_2 < \ldots < j_{\ell-1} < j_\ell$ of positive natural numbers, party p_{j_ℓ} receives a fresh random bit $r_{j_1,j_2,\ldots,j_\ell}$. In addition, for every sequence $j_1 < j_2 < \ldots < j_{k-1} < j_k$ where $\{p_{j_1}, p_{j_2}, \ldots, p_{j_{k-1}}, p_{j_k}\} \in \Gamma$ is qualified, party p_{j_k} receives $s \oplus_{i \in [k-1]} r_{j_1,\ldots,j_i}$. The scheme clearly realizes Γ. Additionally, for every $t \in \mathbb{N}$, the share size of party p_t is at most

$$\sum_{j \leq k-1} \binom{t-1}{j} = O(t^{k-1})$$

. □