Revisiting Beimel-Weinreb Weighted Threshold Secret Sharing Schemes

Oriol Farràs and Miquel Guiot

Universitat Rovira i Virgili, Tarragona, Spain
oriol.farras@urv.cat, miquel.guiot@urv.cat

Abstract. A secret sharing scheme is a cryptographic primitive that allows a dealer to share a secret among a set of parties, so that only authorized subsets of them can recover it. The access structure of the scheme is the family of authorized subsets. In a weighted threshold secret sharing scheme, each party is assigned a weight according to its importance, and the authorized subsets are those in which the sum of their weights is at least the threshold value.
For these access structures, the best general constructions were presented by Beimel and Weinreb [IPL 2006]: The scheme with perfect security has share size $n^{O(\log n)}$, while the scheme with computational security has share size polynomial in n. However, these constructions require the use of shallow monotone sorting networks, which limits their practical use.
In this context, we revisit this work and provide variants of these constructions that are feasible in practice. This is done by considering alternative circuits and formulas for weighted threshold functions that do not require monotone sorting networks. We show that, under the assumption that subexponentially secure one-way functions exist, any WTAS over n parties and threshold σ admits a computational secret sharing scheme where the size of the shares is polylog(n) and the size of the public information is $O(n^2 \log n \log \sigma)$. Moreover, we show that any authorized subset only needs to download $O(n \log n \log \sigma)$ bits of public information to recover the secret.

Keywords: Secret sharing scheme · weighted threshold access structure · threshold cryptography

1 Introduction

A secret sharing scheme is a cryptographic primitive that allows a dealer to share a secret among a set of parties in such a way that only certain subsets of them, called authorized, can recover the secret. The access structure of the scheme is the collection of its authorized subsets. They were introduced independently by Shamir [Sha79] and Blakley [Bla79] in 1979, when they presented schemes for threshold access structures. In these access structures, the authorized subsets are those whose size is at least a given threshold.

While traditional secret sharing schemes are information-theoretic, meaning they remain secure even against adversaries with unbounded computational power, another class of schemes, known as computational secret sharing schemes, achieve security under cryptographic assumptions. These schemes often enable more efficient constructions [Yao89,Kra94,ABI+23], making them particularly appealing for modern applications. Among others, secret sharing schemes are used as a building box of cryptographic protocols for secure multiparty computation and threshold encryption [Bei11]. In many of these applications, schemes with threshold access structures are convenient.

Nevertheless, there are situations that require more general access structures. One of these generalizations is the notion of *weighted threshold access structures* (WTASs). In WTASs, each party is assigned a weight according to its importance, and a subset is authorized if the total weight of its parties meets or exceeds a specified threshold. WTASs are particularly useful in situations where parties have varying levels of significance. For example, in the proof-of-stake model, each validator's stake depends on the amount of cryptocurrency they own, and their influence is proportional to their stake [KRDO17,BCC+21]. Another example arises in the stock exchange, where company shares are distributed non-uniformly among shareholders, and voting power depends on the size of each shareholder's stake. These scenarios highlight the need for secret sharing schemes supporting WTASs.

Despite their utility, constructing efficient secret sharing schemes for WTASs poses several challenges. For these access structures, the share size of the best known information-theoretic schemes is either linear in the weights [Sha79] or quasipolynomial in the number of parties [BW06], often resulting in long shares. On the computational side, Beimel and Weinreb [BW06] presented a scheme with polynomial share size. However, it requires the use of shallow monotone sorting networks, leading to large hidden constants in the share size that limit its use in real-world applications.

In this context, this work revisits the constructions of Beimel and Weinreb [BW06], with special attention to the computational one, with the aim of improving them for practical use.

1.1 Our Results

In this work, we present an improvement of the computational weighted threshold secret sharing scheme proposed by Beimel and Weinreb [BW06] that outperforms the state-of-the-art solutions. In particular, our proposal gets rid off the shallow monotone sorting networks and gives a precise upper bound on the size of the scheme's public information, lowering the existing upper bounds. Our main result is as follows.

Theorem 1.1 (Informal). *Under the assumption that subexponentially secure one-way functions exist, any weighted threshold access structure over n parties with threshold σ admits a computational secret sharing scheme where the size of the shares is* $\mathrm{polylog}(n)$ *and the size of the public information is $\tilde{O}(n^2 \log \sigma)$. Moreover, any authorized subset only needs to download $\tilde{O}(n \log \sigma)$ bits of the public information to recover the secret.*

Taking into account that every WTAS admits weights whose sum is $W = 2^{\lceil n \log n \rceil}$ [Mur71], we obtain the following corollary.

Corollary 1.2 (Informal). *Under the assumption that subexponentially secure one-way functions exist, any weighted threshold access structure admits a computational secret sharing scheme where the size of the shares is* $\mathrm{polylog}(n)$*, the size of the public information is* $\widetilde{O}(n^2)$*, and any authorized subset needs to download* $\widetilde{O}(n)$ *bits of public information.*

Moreover, in most practical applications[Woo14,Yak17,CM17,RYS+20,Lab22], it is sufficient for the weights and threshold to be polynomial in the number of parties. Under this setting, our scheme results in a share size of $\mathrm{polylog}(n)$, a public information size of $\widetilde{O}(n^3)$, and requires the parties to download only $\widetilde{O}(n)$ bits. Our main result and previous best solutions are illustrated in Fig. 1.

	Share Size	**Public Information Size**	**Privacy**
[BW06]	$\mathrm{poly}(n)$	$\mathrm{poly}(n)$	Comp. (Poly. OWF)
[ABI+23,FG24]	$\mathrm{polylog}(n)$	$\mathrm{poly}(n)$	Comp. (Subexp. RSA)
Theorem 1.1	$\mathrm{polylog}(n)$	$O(n^2 \log n \log \sigma)$ $= O(n^3 \log^2 n)$	Comp. (Subexp. OWF)
[Sha79]	$w \log w = 2^{O(n \log n)}$	-	Perfect
[BW06]	$n^{O(\log \log \sigma)} = n^{O(\log n)}$	-	Perfect

Fig. 1: Summary of secret sharing schemes for WTASs. In the second and third columns, we give an upper bound on the share and public information size of the schemes, considering secrets of 1-bit and representing the number of parties by n and the threshold by σ. We use the fact that the best upper bound for the size of the weights in WTASs is $w = 2^{\lceil n \log n \rceil}$ [Mur71], which is tight [Hås94]. In the *Privacy* column, we distinguish between perfect and computational security, and for the later we include the cryptographic assumption.

To obtain the scheme of Theorem 1.1, we use Yao's compiler [Yao89], which reduces the problem of constructing secret sharing schemes for WTASs to the construction of small monotone Boolean circuits realizing monotone weighted threshold functions. This problem has been thoroughly studied in the past, giving positive and negative results [BW06,COS17].

In this context, our main technical contribution is the description of a new monotone Boolean circuit that improves and simplifies the one given by Beimel and Weinreb [BW06] by using threshold gates instead of AND and OR gates.

With that, we can avoid the use of shallow monotone sorting networks and give a precise upper bound on the number of gates and wires of the circuit. Our main technical result is the following.

Theorem 1.3 (Informal). *Any monotone weighted threshold function with n variables and threshold σ can be computed by a monotone Boolean circuit with $O(n \log \sigma)$ gates and $O(n^2 \log \sigma)$ wires.*

Based on this monotone Boolean circuit and combining it with other existing techniques, we obtain further optimizations that lead to more efficient secret sharing schemes for WTASs.

We begin by turning our attention to a recent line of research [BHS23,TF24,FG24] that considers reductions of the weights, i.e. by scaling or rounding them, to construct more efficient schemes for approximate WTASs. We contribute to this question by presenting a new method to reduce the range of the weights. Namely, we adapt our circuit to construct an information-theoretic scheme with logarithmic share size for a rounding approximation of the access structure. Additionally, by combining the results on the approximation of WTASs of Farràs and Guiot [FG24] with our monotone Boolean circuit, we obtain a computational scheme with even smaller public information size at the cost of slightly modifying the original access structure. These results are stated and proved in Section 4.

Later, we can further transform our monotone Boolean circuit into a monotone Boolean formula to obtain a scheme with information-theoretic security that has share size $n^{O(\log \log \sigma)}$, matching the best-known construction [BW06]. Furthermore, we provide a recursive information-theoretic scheme for WTASs. Despite the upper bound on the share size given by this recursive scheme is slightly worse than the one obtained with Shamir's virtualization technique, we provide specific use cases in which it produces considerably smaller shares. These results are deferred to the Appendix.

Open Questions. Our results leave open several questions about the construction of efficient schemes for WTASs. In the following, we highlight four areas where future work could lead to significant improvements of state-of-the-art solutions.

In existing computational schemes for WTASs, the amount of public information required remains supercubic in the number of parties in the worst case. Reducing this to linear or even polylogarithmic would significantly improve the practicality of these schemes. In this context, adapting the construction from Applebaum et al. [ABI+23] to threshold gates would lead to a linear reduction in the size of the public information. Furthermore, for practical applications, leveraging knowledge of the weights distribution could help to develop more efficient schemes.

From an information-theoretic perspective, the share size of the best-known schemes remains quasipolynomial in the number of parties. Moreover, a substantial gap persists between this upper bound and the best-known lower bound of $\Omega(\sqrt{n})$ [PS00], even when restricted to linear schemes. Closing this gap presents a significant challenge: reducing the share size would greatly improve the practicality of these schemes, while strengthening the lower bound would provide deeper insights into the fundamental limitations of these constructions.

In complexity theory, one of the main open questions regarding monotone weighted threshold functions is whether they belong to the class $m\mathcal{NC}^1$. Beimel and Weinreb [BW06] showed that these functions are in $m\mathcal{AC}^1$, but both their circuit and ours require a non-constant fan-in. Notably, constructing an $m\mathcal{NC}^1$ circuit that realizes any weighted threshold function would lead to information-theoretic secret sharing schemes for WTASs with polynomial share size.

Finally, from a broader perspective, the results of [COS17] establish the impossibility of constructing constant-depth, polynomial-size monotone Boolean circuits for weighted threshold functions. This suggests that circuit-based methods alone may be insufficient for designing efficient information-theoretic schemes for these access structures. Consequently, a compelling open question is whether WTAS schemes can be developed using techniques that do not rely on circuits or formulas. Addressing this challenge would likely require novel theoretical tools and unexplored approaches capable of overcoming current limitations.

1.2 Our Techniques

In order to construct efficient secret sharing schemes for WTASs, we perform a two-step procedure. First, we view the access structure as a monotone Boolean function and construct efficient representations of it in terms of monotone majority Boolean circuits, i.e. Boolean circuits using only threshold gates. Later, we apply known cryptographic techniques to convert these alternative representations of our access structure into secret sharing schemes.

Majority Boolean Circuits for Weighted Threshold Functions. Our approach begins with a reduction from any weighted threshold function to a canonical form, where the weights follow a structured pattern of decreasing powers of two. This transformation follows a similar procedure to that of Beimel and Weinreb [BW06]. First, each original weight is decomposed into a sum of distinct powers of two. Then, a new variable is assigned to each of these summands. Finally, additional variables are introduced to ensure that the threshold itself becomes a power of two. This reduction simplifies the subsequent circuit construction by structuring the weights in a way that naturally aligns with binary computations.

With this reduction in place, we construct a polynomial-size monotone majority Boolean circuit that efficiently computes the transformed weighted threshold function. This circuit is designed to perform binary addition using only threshold gates, making it a particularly efficient construction. More specifically, at each level, threshold gates act as counters, summing input variables of specific weights. Their outputs are then fed into the threshold gates of the next level, propagating carry values upward through successive counting stages. Ultimately, if the cumulative weighted sum of the inputs exceeds the threshold, the final 2-threshold gate at the top level outputs a one.

Our circuit can be viewed as a refined and simplified variant of the one proposed by Beimel and Weinreb [BW06], which also computes weighted threshold functions but relies solely on AND and OR gates rather than threshold gates. Our modifications eliminate the need for the sorting networks required in their construction while providing a precise upper bound on the number of gates and wires in the circuit.

Moreover, we further modify the circuit to reduce by a linear factor the public information needed for each authorized subset. In particular, by adding a small number of OR gates and properly wiring the circuit, we can ensure that the reconstruction process only relies on the wires of a single gate per level, significantly reducing the amount of public information needed to recover the secret.

Secret Sharing Schemes Construction. Once we have derived a polynomial-size monotone majority Boolean circuit for WTASs, we still need to construct the information-theoretic and the computational secret sharing schemes. In both cases, we apply already known compilers.

On the computational setting, it suffices to apply Yao's compiler [Yao89] for constructing computational secret sharing schemes from monotone Boolean circuits to our original majority Boolean circuit. In this procedure, the resulting share size is the security parameter, while the size of the public information corresponds to the number of wires of the circuit.

For the information-theoretic scheme, we first apply the same balancing and dynamic programming techniques as Beimel and Weinreb [BW06] to our majority Boolean circuit. This allows us to reduce its depth to logarithmic while preserving its polynomial size, enabling a black-box transformation into a monotone Boolean formula of quasipolynomial size. From there, we apply the technique of Benaloh and Leichter [BL88] for constructing secret sharing schemes from monotone Boolean formulas, where the share size corresponds to the length of the formula. Furthermore, we examine the limitations of this approach in light of the results from [COS17], which establish the impossibility of constructing constant-depth, polynomial-size monotone Boolean circuits for weighted threshold functions. This result diminishes the prospect of achieving a polynomial share size for these access structures through a black-box transformation from monotone Boolean circuits to monotone Boolean formulas.

1.3 Related Work

In this section, we focus on prior work related to the construction of monotone Boolean circuits for monotone weighted threshold functions and the development of secret sharing schemes for WTASs.

Monotone Boolean Circuits for Monotone Weighted Threshold Functions. The study of monotone Boolean circuits consists of understanding the complexity of computing monotone Boolean functions, i.e. Boolean functions that can be expressed using only AND and OR gates. Early foundational work in this area by Razborov [Raz87] and by Alon and Boppana [AB87] established crucial lower bounds for monotone Boolean circuits, revealing that certain monotone Boolean functions, such as clique, require monotone Boolean circuits of exponential size. These results highlighted the limitations of purely monotone constructions for certain classes of Boolean functions.

Goldmann and Karpinski [GK93] advanced this field by proving that weighted threshold gates could be simulated by polynomial-size majority Boolean circuits. However, they found that these simulations were inherently non-monotone, as they introduced negative weights (or, equivalently, negations), even when the target Boolean function itself was monotone. This observation raised the question of whether efficient monotone simulations of monotone weighted threshold functions were possible, particularly in constant depth. Hofmeister [Hof92] provided some progress by demonstrating that monotone, depth-2 Boolean circuits for specific monotone weighted threshold functions would require exponential size, emphasizing the limitations of low-depth monotone Boolean circuits.

In an independent line of work, Beimel and Weinreb [BW06] constructed polynomial-size logarithmic-depth monotone Boolean circuits for any weighted threshold function. Recently, this construction has been proved to be asymptotically optimal, since Chen, Oliveira and Servedio [COS17] answered negatively the question posed by Goldmann and Karpinski [GK93] by proving that there exists a monotone weighted threshold function such that any constant depth monotone circuit realizing it requires exponential size. Furthermore, they also provide a description of a polynomial-size logarithmic-depth majority Boolean circuit realizing monotone weighted threshold functions.

Secret Sharing Schemes for Weighted Threshold Access Structures. In 1979, Shamir [Sha79] and Blakley [Bla79] introduced the first secret sharing schemes for threshold access structures. Shamir also proposed a method for realizing WTASs using threshold schemes through virtualization: given weights w_i and a threshold σ, the dealer treats party i as w_i distinct parties, distributing w_i separate shares of the σ-threshold scheme to that party. This approach results in a total share size of $O(W \log W)$ for any access structure, where W is the total sum of the weights. However, in the worst-case scenario, i.e. when the weights are exponential in the number of parties [Hås94], this scheme produces larger shares than those of the best known construction for general access structures.

Beimel and Weinreb [BW06] presented new constructions that are based on a polynomial-size logarithmic-depth monotone Boolean circuit that realizes these access structures. Then, they apply Yao's technique [Yao89] for monotone circuits to obtain a scheme that is computationally secure. Namely, under the assumption of the existence of secure one-way functions, the scheme realize the access structure with shares and public information of polynomial size in the number of parties. Furthermore, under the stronger assumption that subexponentially secure one-way functions exist, the share size in their construction can be further reduced to be polylogarithmic in the number of parties.

Converting this circuit into a monotone boolean formula and applying Benaloh and Leichter [BL88], Beimel and Weinreb [BW06] provide an information-theoretic scheme for WTASs. Implicitly, they show that any WTAS with threshold σ admits a scheme with share size $n^{O(\log\log\sigma)}$. Since the best upper bound for the threshold is $\sigma = 2^{O(n\log n)}$, it implies that every WTAS admits a perfect scheme of share size $n^{O(\log n)}$. This is the best general construction known to date.

This result highlights that WTASs occupy a *privileged* position in terms of efficiency, as most general access structures require linear schemes with a normalized share size of $2^{n/3+o(n)}$ [BF20]. The best-known lower bound on the information ratio is $\Omega(\sqrt{n})$, established by Padró and Sáez [PS00] through an analysis of WTASs with only two distinct weights. This also remains as the best lower bound on share size to date. Characterizations of ideal WTASs can be found in [BTW08,FP12]. For a more comprehensive overview of previous results on weighted and hierarchical access structures, see [FG24].

Recently, Benhamouda, Halevi, and Stambler [BHS23], Garg et al [GJM$^+$23], and Tonkikh and Freitas [TF24] explored a relaxed weighted threshold model, considering ramp WTASs. In these access structures, there are privacy and correctness thresholds, and each party has a single weight. This setting is less restrictive and admits schemes that are not perfect, allowing a reduction of the share size. Similarly, Farràs and Guiot [FG24] introduced a technique to approximate any WTAS by another one with small weights. In [FG24], there is also a computational scheme for WTASs that has polylogarithmic share size, and polynomial public information size. This scheme is a direct instantiation of the general construction of Applebaum et al. [ABI$^+$23] to the weighted threshold case.

1.4 Organization

In Section 2, we lay out the preliminaries on the complexity of Boolean functions and secret sharing schemes. In Section 3, we construct the polynomial-size majority Boolean circuit of Theorem 1.3 and we prove Theorem 1.1. Finally, Section 4 focuses on the schemes for approximate access structures. We defer to the Appendix illustrative examples of our circuit (Appendix A), the information-theoretic scheme that results from transforming our circuit into a monotone formula (Appendix B), and a recursive construction (Appendix C).

2 Preliminaries

Notation. We notate $\mathbb{N}$ and $\mathbb{R}$ for the sets of the non-negative integer and real numbers, respectively. For $n \in \mathbb{N}$, we denote the set $\{1,\dots,n\}$ as $[n]$. For a set S, we denote its cardinal as $|S|$. We denote vectors $\boldsymbol{x}$ using bold symbols, their i-th coordinates as x_i, and their support as $\mathrm{supp}(\boldsymbol{x}) = \{i \in [n] \,:\, x_i \neq 0\}$. The unary vector is denoted by $\mathbf{1}^n \in \mathbb{R}^n$ and the zero vector is denoted by $\mathbf{0}^n$. For any vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, we denote their concatenation as $\boldsymbol{x}||\boldsymbol{y}$, and we say that

$\boldsymbol{x} \leq \boldsymbol{y}$ if and only if $x_i \leq y_i$ for all $i \in [n]$. For a vector $\boldsymbol{x} \in \mathbb{R}^n$ with indices in the set $[n]$, for every $A \subseteq [n]$ we define $\boldsymbol{x}_A$ as the vector whose elements are the i-th coordinates of $\boldsymbol{x}$ with $i \in A$.

Next, we define thre functions over the reals. We set sign: $\mathbb{R} \to \{0,1\}$ as the function with $\text{sign}(x) = 1$ if and only if $x \geq 0$; for $\boldsymbol{x}, \boldsymbol{w} \in \mathbb{R}^n$ and $\sigma \in \mathbb{R}$, we set $\text{WTF}(\boldsymbol{w}, \sigma)(\boldsymbol{x}) = \text{sign}(\boldsymbol{w} \cdot \boldsymbol{x} - \sigma)$; and for $\boldsymbol{x} \in \mathbb{R}^n$ and $\sigma \in \mathbb{R}$, we set $\text{Th}(\sigma)(\boldsymbol{x})$ as the σ-threshold function.

2.1 Complexity of Boolean Functions

In this section, we present the necessary definitions and technical results on the complexity of Boolean functions. Specifically, we focus on the notion of weighted threshold function and its properties.

Weighted Threshold Functions. We start by defining the main block of our construction: weighted threshold functions.

Definition 2.1 (Weighted Threshold Function). *Let $\boldsymbol{w} \in \mathbb{R}^n$ and $\sigma \in \mathbb{R}$. A Weighted Threshold Function (WTF) is a Boolean function $f : \{0,1\}^n \to \{0,1\}$ of the form $f(\boldsymbol{x}) = \text{WTF}(\boldsymbol{w}, \sigma)(\boldsymbol{x})$. The vector $(\boldsymbol{w}, \sigma)$ is said to represent f, while the vector $\boldsymbol{x} \in \{0,1\}^n$ is said to be* authorized *(resp.* forbidden*) if $f(\boldsymbol{x}) = 1$ (resp. $f(\boldsymbol{x}) = 0$).*

To simplify the notation, from now on we will assume that any WTF has the weights sorted in decreasing order. Among them, we are interested in those that are monotone because they are in one-to-one correspondence with WTASs. In this regard, the following remark gives a characterization of monotone WTFs.

Remark 2.2. A WTF given by $f(\boldsymbol{x}) = \text{WTF}(\boldsymbol{w}, \sigma)(\boldsymbol{x})$ is monotone if and only if there exists a representation of f with $w_i \geq 0$ for any $i \in [n]$. Indeed, if f is monotone increasing and $w_i < 0$, then f does not depend on x_i, and we can set its weight to 0.

An important result about monotone WTFs is that they can be computed by polynomial-size logarithmic-depth monotone circuits of unbounded fan-in [BW06]. Furthermore, it is also known that not all monotone WTFs can be computed by polynomial-size constant depth monotone circuits of unbounded fan-in [COS17].

Theorem 2.3 ([BW06,COS17]). *The family of weighted threshold functions is contained in $m\mathcal{AC}^1$ but it is not in $m\mathcal{AC}^0$.*

Next, we state two useful results about the weights of monotone WTFs. The first one is an upper bound on the size of the weights of any monotone WTF. The second one shows that this upper bound is indeed tight.

Theorem 2.4 ([Mur71]). *For any monotone WTF f with n variables, there exist $w_1, \ldots, w_n, \sigma \in \mathbb{N}$ smaller than $2^{\lceil n \log n \rceil}$ such that $f(\boldsymbol{x}) = \text{WTF}(\boldsymbol{w}, \sigma)(\boldsymbol{x})$, where $\boldsymbol{w} = (w_1, \ldots, w_n)$.*

Theorem 2.5 ([Hås94]). *For any $n \in \mathbb{N}$ there exists a monotone WTF with n variables requiring weights of size $2^{O(n \log n)}$.*

We end this section by introducing the concept of distance between Boolean functions, which will be usefull when dealing with the approximation of weighted threshold access structures.

Definition 2.6 (Function Distance). *The distance between two Boolean functions f, g is defined as $\mathrm{dist}(f, g) = \mathbf{E}[|f(\boldsymbol{x}) - g(\boldsymbol{x})|]$. If $\mathrm{dist}(f, g) < \varepsilon$ we say that f and g are ε-close.*

Recently, Farràs and Guiot [FG24] proved that any monotone WTF can be approximated by another monotone WTF whose weights have almost linear size in the number of variables. Their statement follows.

Theorem 2.7 ([FG24]). *For any monotone WTF f over n variables there exists another montone WTF $g = \mathrm{WTF}(\boldsymbol{w}, \sigma)(\boldsymbol{x})$ over n variables such that f and g are $o(1)$-close and $\|\boldsymbol{w}\| = n^{1+o(1)}$.*

2.2 Secret Sharing Schemes

In the following, we present the basics of secret sharing schemes. For convenience, we work with access structures described by monotone Boolean functions, which is equivalent to work with monotone increasing families of subsets.

Definition 2.8 (Access Structure). *An n-party access structure is a monotone Boolean function $f : \{0,1\}^n \to \{0,1\}$ such that $f(\mathbf{0}^n) = 0$ and $f(\mathbf{1}^n) = 1$.*

If $f(\boldsymbol{x}) = 1$, we say that the set $A = \mathrm{supp}(\boldsymbol{x})$ is authorized*, otherwise we say that A is* forbidden*.*

Definition 2.8 implies that non-constant monotone WTF are a particular case of access structures. In this context, they are also called *weighted threshold access structures* (WTASs).

Remark 2.9. In the case of a WTAS given by $f(\boldsymbol{x}) = WTF(\boldsymbol{w}, \sigma)(\boldsymbol{x})$, the family of authorized subsets corresponds to

$$\Gamma = \left\{ A \subseteq [n] \ : \ \sum_{i \in A} w_i \geq \sigma \right\}.$$

We now state the definitions of information-theoretic and computational secret sharing schemes.

Definition 2.10 (Information-Theoretic Secret Sharing Scheme). *Let $\mathcal{S}$ be a finite set and let f be an access structure over n parties. A secret sharing scheme for f is a pair of a randomized algorithm* Share *and deterministic algorithms* $\mathsf{Reconstruct}$ *such that*

- ***Perfect correctness.*** *For any secret $s \in \mathcal{S}$ and any authorized set A, it holds that*
$$\mathbf{P}[s = \mathsf{Reconstruct}(\mathsf{Share}(s)_A)] = 1,$$
where $\mathsf{Share}(s)_A$ denotes the restriction of the output of $\mathsf{Share}(s)$ to the parties in A.
- ***Perfect privacy.*** *For any secrets $s, s' \in \mathcal{S}$, any forbidden set B, and any possible set of shares $\{s_i\}_{i\in B}$, it holds that*
$$\mathbf{P}[\{s_i\}_{i\in B} = \mathsf{Share}(s)_B] = \mathbf{P}[\{s_i\}_{i\in B} = \mathsf{Share}(s')_B].$$

Definition 2.11 (Computational Secret Sharing Scheme,[ABI$^+$23]). *Let $\mathcal{S}$ be a finite set, let $\lambda \in \mathbb{N}$ be the security parameter, and let f be an access structure over n parties. A computational secret sharing scheme for f is a pair of a randomized polynomial-time algorithm* Share *and a deterministic polynomial-time algorithm* Reconstruct *such that*

- ***Correctness.*** *For any secret $s \in \mathcal{S}$, any authorized set A, and a description D of f it holds that*
$$\mathbf{P}[s = \mathsf{Reconstruct}(\mathsf{Share}(s, D, \mathbf{1}^\lambda)_A, D)] = 1,$$
where $\mathsf{Share}(s, D, \mathbf{1}^\lambda)_A$ denotes the restriction of the output of $\mathsf{Share}(s, D, \mathbf{1}^\lambda)$ to the parties in A.
- ***Privacy.*** *The scheme is $t(\lambda)$-secure if any $t(\lambda)$-time adversary wins the security game of Definition 2.12 with probability at most $\frac{1}{t(\lambda)}$.*

The security of computational secret sharing schemes is given in terms of a game between an adversary and a challenger. The definition follows.

Definition 2.12 (Security,[ABI$^+$23]). *Let $\mathcal{S} = \{0, 1\}$, let $\lambda \in \mathbb{N}$ be the security parameter, let f be an access structure over n parties, and let* (Share, Reconstruct) *be a computational secret sharing scheme for f. Consider the following game between a non-uniform $t(\lambda)$-time adversary $\mathcal{A}$ and a challenger:*

1. *The adversary $\mathcal{A}$ on input $\mathbf{1}^\lambda$ chooses a forbidden set A and sends it to the challenger.*
2. *The challenger chooses a uniformly random secret $s \in \mathcal{S}$. It computes $\mathsf{Share}(s, D, \mathbf{1}^\lambda)$ and sends $\mathsf{Share}(s, D, \mathbf{1}^\lambda)_A$ to the adversary.*
3. *The adversary outputs a value $s' \in \mathcal{S}$.*

The adversary wins the game if $s' = s$.

We say that the computational secret sharing scheme is $t(\lambda)$-secure if for every non-uniform $t(\lambda)$-time adversary $\mathcal{A}$ and sufficiently large λ, the probability that $\mathcal{A}$ wins is at most $\frac{1}{2} + \frac{1}{t(\lambda)}$. By default, we require $t(\lambda)$-security for every polynomial $t(\cdot)$. In any case, we always assume that $t > \lambda$.

3 Computational Weighted Threshold Secret Sharing Schemes

In this section, we present an improved version of the computational weighted threshold secret sharing scheme proposed by Beimel and Weinreb [BW06]. The main result is the following.

Theorem 3.1 (Theorem 1.1 restated). *Let $\lambda \in \mathbb{N}$ be the security parameter. Under the assumption that polynomially (resp. subexponientally) secure one-way functions exist, any WTAS over n parties and threshold σ admits a computational secret sharing scheme where the size of the shares is $\mathrm{poly}(n, \lambda)$ (resp. $\mathrm{poly}(\log n, \lambda)$) and the size of the public information is $O(n^2 \lambda \log \sigma)$. Moreover, any authorized subset only needs to download $O(n\lambda \log \sigma)$ bits of the public information to recover the secret.*

We divide the proof of Theorem 3.1 into three steps. First, we present a reduction from any WTF to a canonical WTF in which the weights are structured as decreasing powers of two. Next, we construct a polynomial-size monotone majority Boolean circuit for this canonical WTF and use it to derive our computational scheme. Finally, we show how to further reduce the public information needed for the secret reconstruction.

3.1 On the Reduction between Monotone Weighted Threshold Functions

In order to construct a monotone majority Boolean circuit for any monotone WTFs, we first present a reduction from any WTF to a canonical WTF in which the weights are structured as decreasing powers of two. This reduction simplifies the subsequent construction of the circuit by aligning the weights with a structure naturally suited for binary computations.

To do so, we follow an idea similar to the one in [BW06], where they perform a reduction from any monotone WTF to a so-called *universal* monotone WTF. In more detail, given any monotone WTF f over n variables, we convert it into a monotone WTF g over $m = O(n \log \sigma)$ variables such that its threshold and all its weights are decreasing powers of two, and any authorized input $x \in \{0,1\}^n$ for f can be converted to an authorized input $y \in \{0,1\}^m$ for g. The following theorem formalizes this reduction.

Theorem 3.2. *Let $f(\boldsymbol{x}) = \mathrm{WTF}(\boldsymbol{w}, \sigma)(\boldsymbol{x})$ be a monotone WTF over n variables. Then there exists $\sigma' < 2\sigma$, a monotone decreasing sequence of $m = O(n \log \sigma)$ powers of two $\boldsymbol{v}$, and an injective mapping $\rho : \{0,1\}^n \to \{0,1\}^m$ for which the monotone WTF $g(\boldsymbol{x}) = \mathrm{WTF}(\boldsymbol{v}, \sigma')(\boldsymbol{x})$ satisfies that $f(\boldsymbol{x}) = g(\rho(\boldsymbol{x}))$.*

Proof. Without loss of generality, we can assume that the threshold σ is greater than all the weights. Therefore, each weight w_i can be expressed as a summation of at most $\lceil \log \sigma \rceil$ distinct powers of two. From there, for any $i \in [n]$ let $S_i \subseteq [\lceil \log \sigma \rceil]$ be the set such that $w_i = \sum_{j \in S_i} 2^j$ and S_σ be the set such that $\sigma = \sum_{j \in S_\sigma} 2^j$. We construct g and ρ as follows:

1. For each variable x_i with weight w_i of f, the new function g has $|S_i|$ variables $y_{i,j}$ with weights $v_{i,j} = 2^j$ for $j \in [S_i]$.
2. Let k be the integer that satisfies $2^{k-1} < \sigma < 2^k$. Then we set the threshold σ' of g to be 2^k.
3. For each $j \in [k] \setminus S_\sigma$, we add a new constant $y_{\sigma,j} = 1$ with weight $v_{\sigma,j} = 2^j$ to g.
4. We set $g'(\boldsymbol{y}) = \mathrm{WTF}(\boldsymbol{v}, \sigma')(\boldsymbol{y})$ where $\boldsymbol{v}$ is the vector containing $v_{i,j}$ for $i \in [n] \cup \sigma$ and $j \in S_i$.
5. Then, ρ' maps any $\boldsymbol{x} \in \{0,1\}^n$ to $\boldsymbol{y} \in \{0,1\}^m$ with $y_{i,j} = 1$ if and only if $x_i = 1$ or $i = \sigma$.
6. Finally, g and ρ simply consist in a reordering of g' and ρ' to ensure that all the weights appear in decreasing order.

From there, ρ is injective by definition and it is straightforward to check that $f(\boldsymbol{x}) = 1$ if and only if $g(\rho(\boldsymbol{x})) = 1$. □

Intuitively, the reduction performed in Theorem 3.2 consists of decomposing each original weight as the sum of the various powers of two that add up to it, and then associating a new variable to each of these summands. Finally, we must add some constants set to one to impose that the threshold is the largest power of two among all of them without changing the structure of the function. By performing this reduction, we are translating the problem of computing any monotone WTF to the problem of testing whether the sum of a given set of powers of two is greater than an even greater power of two. In practical terms, this reshaping of the weights leads to a structure that is inherently compatible with binary calculations, which facilitates the circuit construction.

3.2 Computational Secret Sharing Schemes for Weighted Threshold Access Structures

Once we have shown how to reduce the computation of any monotone WTF to the computation of the so-called *universal* monotone WTF through Theorem 3.2, we move to construct a monotone majority Boolean circuit for it.

Before formally constructing the circuit, we give a high-level overview of the idea behind it. Intuitively, since by Theorem 3.2 we can assume that the threshold σ and the weights are decreasing powers of two ranging from 1 to $2^{\lceil \log \sigma \rceil}$, the circuit simply performs binary addition and checks whether the resulting value is above the threshold. In more detail, the threshold gates of each level act as a counting machine that calculates how many variables with a given weight appear at the input. Then, appropriately taking their output wires as inputs for the threshold gates of the level above, the circuit converts the previous sum into the carry value to be used in the counting process of the next level. In this way, whenever the sum of the weights of the input variables is above the threshold, the unique 2-threshold gate at the top level of the circuit will evaluate to one.

The general construction is presented in Theorem 3.3 and it is outlined in Fig. 2. See Appendix A.1 for a concrete instantiation of the circuit.

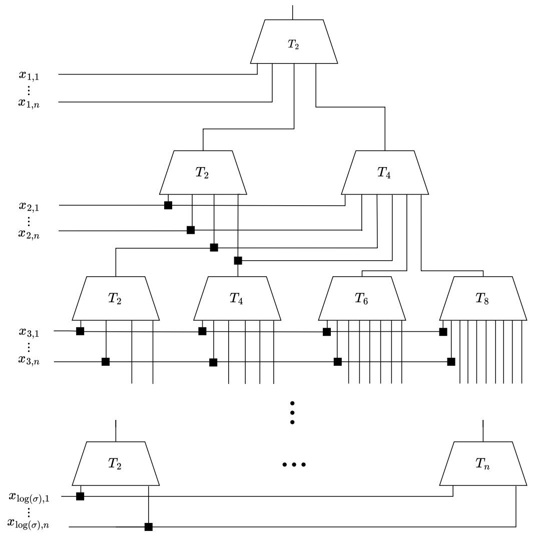

Fig. 2: Monotone majority Boolean circuit realizing a WTAS over n variables.

Theorem 3.3 (Theorem 1.3 restated). *Any monotone WTF over n variables and threshold σ can be computed by a majority Boolean circuit with $O(n \log \sigma)$ gates and $O(n^2 \log \sigma)$ wires.*

Proof. Let f be a monotone WTF over n variables and threshold σ. First, recall that by using the reduction from Theorem 3.2 we simply need to build a majority Boolean circuit for the monotone WTF g consisting of $O(n \log \sigma))$ variables and in which the threshold and the weights are consecutive decreasing powers of two. In particular, the threshold is set to $\sigma' = 2^{\lceil \log \sigma \rceil}$ and for each weight w_i there are at most n distinct variables.

We now move to describe the circuit and to prove its correctness. First, recall that as explained in Section 3.1, the computation of g can be interpreted as simply checking if the addition of the powers of two associated with the input variables reaches the power of two set as the threshold, i.e. it can be viewed as computing binary addition.

With this in mind, we split the variables per levels according to their weight, being the top level the one containing the variables of weight $\frac{\sigma'}{2}$ and the lowest level the one containing the variables of weight one. At each level i, we add a threshold gate for each even value ranging from two to the minimum between $2n$ and $\frac{\sigma'}{w_i}$. In this regard, the $2n$ upper bound limits the amount of values that can appear per level (there are at most n variables of any weight plus at most n carry values), and the $\frac{\sigma'}{w_i}$ upper bound limits the amount of values that we need to add (if in the level i there are more than $\frac{\sigma'}{w_i}$ variables in the input, the threshold is trivially satisfied).[1]

The goal of this threshold gates is to count how many variables of a given level appear in the input and convert them into variables of the level above to mimic the notion of the carry bits. Moreover, all the output wires of these threshold gates are connected as input wires to the threshold gates of the level above to make them count as carry bits.[2] The output of the circuit simply consists in the output wire of the unique 2-threshold gate at the top level.

To prove its correctness, we follow the same argument as in the work of Beimel and Weinreb [BW06]. For every $i \in [\log \sigma']$, let a_i be the number of variables of level i that appear in the input, let $v_i = \sum_{j \in [i]} 2^{j-1} a_j$, and let c_i be the carry value propagated by the level i to the level $i+1$. We set by definition $c_0 = v_0 = 0$.

Claim. For every $0 \leq i \leq \log \sigma'$ it holds that $v_i - 2^i < 2^i c_i \leq v_i$.

Proof of the Claim. We prove the claim by induction. The base case $i = 0$ is correct by inspection. To prove the inductive step, we first notice that by definition it holds that

$$v_i = v_{i-1} + 2^{i-1} a_i.$$

Now, using the induction hypothesis, the first inequality holds because

$$2^i c_i = 2^i \left\lfloor \frac{a_i + c_{i-1}}{2} \right\rfloor \geq 2^i \frac{a_i + c_{i-1} - 1}{2} = 2^{i-1} a_i + 2^{i-1} c_{i-1} - 2^{i-1} > v_i - 2^i.$$

Finally, using again the induction hypothesis, the second inequality holds because

$$2^i c_i = 2^i \left\lfloor \frac{a_i + c_{i-1}}{2} \right\rfloor \leq 2^i \frac{a_i + c_{i-1}}{2} = 2^{i-1} a_i + 2^{i-1} c_{i-1} \leq 2^{i-1} a_i + v_{i-1} = v_i.$$

□

Now, to prove the correctness of the circuit it suffices to notice that whenever $v_{\log \sigma'} \geq \sigma'$, the first inequality of the Claim implies that $c_{\log \sigma} = 1$, so the circuit accepts. While when $v_{\log \sigma'} < \sigma'$, the second inequality of the Claim implies that $c_{\log \sigma} = 0$, so the circuit rejects.

It only remains to prove the bound on the number of gates and wires of the circuit. From its description, it is clear that the circuit has at most $\log \sigma' = O(\log \sigma)$ levels. In each level there are at most n distinct input variables and n distinct carry values, which limits the number of gates per level to n. Hence,

[1] At the lowest level, since there is no carry value the upper range of the threshold is simply n.

[2] In fact, the output wires of a t-threshold gate only need to be connected as input wires of the threshold gates of the level above whose threshold is greater or equal than $\frac{t}{2}$. However, since this optimization does not improve the overall complexity of the circuit, we omit the details.

there are at most $O(n \log \sigma)$ gates in the circuit. Moreover each gate has at most $2n$ input wires, which correspond to the n variables and n carry values per level. Therefore, there are at most $O(n^2 \log n)$ wires in the circuit. $\square$

Remark 3.4. Notice that in the proof of Theorem 3.3, if instead of taking the threshold σ as a parameter we replace it by its upper bound of $O(n \log n)$ given by Theorem 2.4, the depth of the circuit can be upper bounded by $O(n \log n)$. This leads to the upper bounds of $O(n^2 \log n)$ and $O(n^3 \log n)$ for the number of gates and wires, respectively.

Once we have constructed a polynomial-size majority Boolean circuit for any WTF, we simply apply Yao's compiler [Yao89] to construct a computational secret sharing scheme for WTASs.

Proposition 3.5. *Let $\lambda \in \mathbb{N}$ be the security parameter. Under the assumption that polynomially (resp. subexponientally) secure one-way functions exist, any WTAS over n parties and threshold σ admits a computational secret sharing scheme where the size of the shares is* $\text{poly}(n, \lambda)$ *(resp.* $\text{poly}(\log n, \lambda)$*) and the size of the public information is $O(n^2 \lambda \log \sigma)$.*

Proof. We apply Yao's compiler [Yao89] to the majority Boolean circuit of Theorem 3.3 to obtain the desired computational secret sharing scheme. In such construction, when using threshold gates the size of the public information corresponds to the product of the number of wires and the size of the data transmitted through them. Moreover, the share size corresponds to the security parameter, which is polynomial or polylogarithmic in the number of parties depending on the security assumption on the one-way functions. $\square$

Remark 3.6. As in Remark 3.4, an alternative upper bound on the public information is $O(n^3 \lambda \log n)$.

3.3 Reduction of the Public Information

We now show how to reduce the amount of public information needed for each authorized subset during the reconstruction process by a linear factor at the cost of increasing its total size by a constant factor. To do so, we slightly modify the circuit constructed in Section 3.2 by adding to it $O(n \log \sigma)$ additional OR gates and $O(n^2 \log \sigma)$ new wires.

Prior to delving into the formal proof, we outline its core idea. The circuit, as constructed in Theorem 3.3, requires the parties to use the wires of all gates whose output is set to 1 to recover the secret. However, this approach is inefficient in terms of the number of gates used. Essentially, at any given level, among all gates whose output is set to 1, only the gate with the highest threshold matters, as it dominates the others. Specifically, if the output of a gate is 1, then all gates with lower thresholds at that level also have their output set to 1.

To optimize this, we ensure that each authorized subset only considers the wires of these critical gates. This is achieved by connecting the output wire

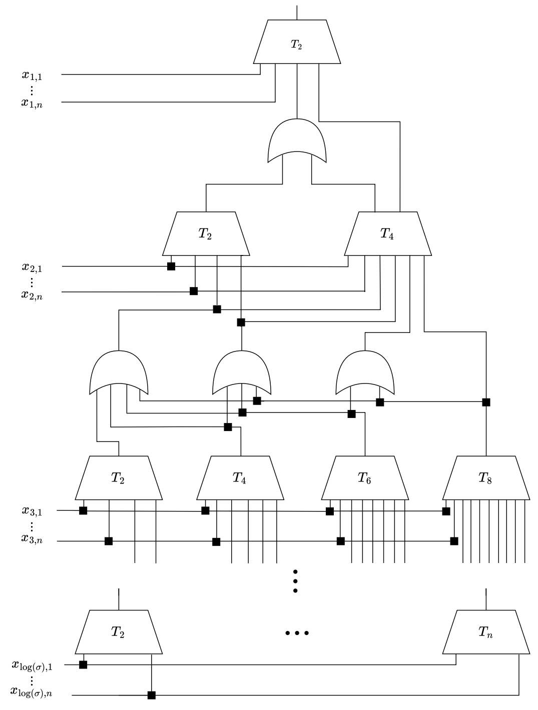

Fig. 3: Optimized version of the circuit realizing a WTAS over n variables.

of each threshold gate to an OR gate, which also receives as input the output wires of all threshold gates with higher thresholds at the same level. In this way, during the reconstruction process, at any level it suffices to look at the public information associated to the wires of a single gate rather than that of all n gates.

Fig. 5 corresponds to the optimization of the circuit of Fig. 4 and Appendix A.2 contains a concrete instantiation. The result is stated as follows.

Proposition 3.7. *At the cost of increasing the size of the public information in Proposition 3.5 by a constant factor, the amount of public information that each authorized subset needs to download during the reconstruction process is reduced to $O(n\lambda \log \sigma)$.*

Proof. Let f be a monotone WTF over n variables and threshold σ, and let C be the monotone majority Boolean of Theorem 1.1 computing it. Due to Yao's compiler [Yao89], to prove the result it suffices to build another circuit C' that computes f with the same complexity as C and then argue that the amount of wires needed for any authorized subset to recover the secret during the reconstruction phase is $O(n \log \sigma)$.

To construct the circuit C', we take the circuit C as the starting point. For any level $i \in [O(\log\sigma)]$, we first connect the output wire of each threshold gate G to an OR gate G'. Then, each of these OR gates G' also takes as input the output wires of all threshold gates at level i with thresholds higher than that of G. Finally, the output wire of G' replaces the original output wire of G as input to all the gates at level $i+1$.

Once the circuit C' is built, we check its correctness. To do so, it suffices to notice that the output of any of those new OR gates is set to 1 if and only if the output of the smallest threshold gate connected to it is set to 1. In fact, this holds because of the correctness of the circuit C.

Now, we move to prove the bound on the number of gates and wires of the circuit C'. By construction, it is clear that its depth is twice the depth of C, i.e. it is upper bounded by $O(\log\sigma)$. Moreover, there are at most $n-1$ new OR gates and the same amount of threshold gates per level. Finally, each of these new OR gates has at most n output wires, and at most n output wires are added to each of the threshold gates. This results in upper bounds of $O(n\log\sigma)$ gates and $O(n^2\log\sigma)$ wires for C'.

To conclude, it only remains to prove that the amount of wires that each authorized subset needs to use during the reconstruction phase is $O(n\log\sigma)$. By construction, at each level $i \in [O(\log\sigma)]$ it suffices to look at a single threshold gate. Therefore, any authorized subset only needs to look at both the n input wires and the n output wires of these gates. This gives a total amount of $O(n\log\sigma)$ wires. $\square$

From a theoretical perspective, the circuit constructed in Proposition 3.7 is worse than that in Theorem 3.3, since it uses around twice as many gates and wires to compute the same function. However, from a practical approach, it is more efficient because it allows any authorized subset to recover the secret using less data and computation. In this sense, it is useful to think of public information as a huge dataset that the dealer publishes during the sharing phase over the internet. For example, the dealer uploads it on a server. With this in mind, the circuit constructed in Proposition 3.7 allows any authorized subset of parties to download, during the reconstruction phase, only the public information corresponding to the wires of a single gate per level, which reduces the data they must handle by a linear factor. In addition, this also reduces by the same proportion the amount of computations the parties need to perform, since they only need to compute one polynomial interpolation per level.

3.4 Comparison with State-of-the-Art Proposals

We compare our proposal with two previous computational schemes: The scheme by Beimel and Weinreb [BW06], and the scheme by Farràs and Guiot [FG24], which is a direct instantiation of the general construction of Applebaum et al. [ABI+23] to the weighted threshold case. This information is summarized in Fig. 1.

The primary distinction between the result in Theorem 3.1 and the previous computational schemes [BW06,FG24] lies in the design of the underlying monotone Boolean circuits used to compute monotone WTF. Specifically, their constructions rely on circuits that perform binary addition using only AND and OR gates, combined with shallow monotone sorting networks. However, these sorting networks are intricate to describe and implement [AKS83,Val84], and their complexity hides huge constants that make them impractical for real-world use. In contrast, our approach eliminates the need for these sorting networks by using threshold gates to perform binary addition directly. This produces a monotone majority Boolean circuit that not only simplifies but also enhances the design proposed by Beimel and Weinreb, making it far more suitable for practical applications.

Furthermore, although the depth of our circuit is larger by a linear factor compared to the depth of the circuit by Chen, Oliveira and Servedio [COS17] ($O(n \log n)$ versus $O(\log n)$ respectively), our circuit has a significantly smaller overall size ($O(n^2 \log n)$ versus $O(n^{18} \log^6 n)$). Consequently, using our circuit as a building block for the scheme results in a substantial reduction in the size of public information. See Remark B.1 for further details.

In terms of share size, the schemes in Theorem 3.1 and [FG24] achieve polylogarithmic share size, while the one in [BW06] has polynomial share size. Nevertheless, under the stronger assumption of subexponentially secure one-way functions, the scheme of Beimel and Weinreb [BW06] could also achieve polylogarithmic share size, as it relies on the same monotone Boolean circuit for WTFs used in [FG24].

Our construction provides an explicit upper bound on the size of public information, expressed as a polynomial of a specific degree. This advantage arises from the simplicity of our monotone majority Boolean circuit, which allows for straightforward counting of gates and wires. Moreover, the optimization presented in Proposition 3.7 further enhances the efficiency of public information in our scheme, amplifying the gap between our approach and others and making our construction more suitable for practical implementations. In contrast, the scheme in [FG24] has public information size that scales linearly with the number of gates, while both Theorem 3.1 and [BW06] show a linear scaling with the number of wires.

Another key difference lies in the cryptographic assumptions. Our proposal and the scheme in [BW06] rely on the existence of secure one-way functions, whereas the construction in [FG24] is based on the stronger RSA assumption. This divergence arises from the use of different secret sharing compilers for monotone Boolean circuits: The first ones employ Yao's technique [Yao89], while the second one is built on results from Applebaum et al. [ABI$^+$23].

4 Schemes for Approximate Weighted Threshold Access Structures

A current line of research focuses on reducing weights of WTASs in order to construct more efficient schemes [BHS23,TF24,FG24]. By doing so, the resulting secret sharing schemes realize WTASs that approximate the given ones.

In this section, we explore how our proposal can benefit from these advancements. First, in Theorem 4.1, we leverage the work of Farràs and Guiot [FG24] to further minimize the size of the public information in our scheme. Subsequently, in Theorem 4.2 we adapt our circuit to construct an information-theoretic scheme with logarithmic share size for a rounding approximation of the access structure.

By combining Theorem 2.7 with our majority Boolean circuit, we can reduce the size of the public information in our computational scheme, with only a slight modification to the original access structure. Specifically, by applying Theorem 2.7 to the monotone WTF that defines the access structure, we obtain a new monotone WTF that is $o(1)$-close to the original, with a maximum weight upper bounded by $n^{1+o(1)}$. As a result, the depth of the monotone majority Boolean circuit realizing this new monotone WTF is reduced to $O(\log n)$, implying that it only has $O(n \log n)$ gates and $O(n^2 \log n)$ wires. From there, using Yao's technique [Yao89] we obtain the following result.

Theorem 4.1. *Under the assumption that subexponentially secure one-way functions exist, for any weighted threshold access structure Γ over n parties there exists another weighted threshold access structure $o(1)$-close to Γ that admits a computational secret sharing scheme where the size of the shares is* $\mathrm{polylog}(n)$ *and the size of the public information is $O(n^2 \lambda \log n)$.*

On the information-theoretic side, to achieve a scheme with logarithmic share size, we can constrain the depth of our majority Boolean circuit to a constant by effectively "cutting" it. Intuitively, this process corresponds to a rounding procedure applied to the function's weights, reducing them to a constant while incurring a slight loss in resolution.

Theorem 4.2. *Let Γ be a weighted threshold access structure over n parties with threshold σ and weights $\boldsymbol{w}$. For any $c \in [\log \sigma]$, there exists a weighted threshold access structure $\Gamma' \subseteq \Gamma$ with threshold σ' and weights $\boldsymbol{w}'$ satisfying*

1. *$\sigma \leq \sigma' < 2\sigma$,*
2. *for any $A \notin \Gamma$ it holds that $A \notin \Gamma'$,*
3. *for any $A \in \Gamma$ such that $\sum_{i \in A} w'_i \geq \sigma' + \frac{n\sigma'}{2^c}$ it holds that $A \in \Gamma'$*

that admits a secret sharing scheme with share size $O(\log n)$.

Proof. Consider the majority Boolean circuit for Γ as defined in Theorem 3.3. By truncating it at depth c, we obtain a circuit representation of a WTAS Γ' that naturally satisfies conditions (1), (2), and (3). Furthermore, both the threshold and weights of Γ' are multiples of $2^{\lceil \log \sigma \rceil - c}$, so by dividing them by this factor, we can reduce their maximum size to at most 2^c. Consequently, applying Shamir's virtualization technique to Γ' results in a scheme with a share size of $2^c \log n = O(\log n)$. $\square$

Acknowledgments

We thank Amos Beimel for valuable comments and suggestions.

The authors are supported by grant 2021 SGR 00115 from the Government of Catalonia, by the project ACITHEC PID2021-124928NB-I00 funded by MCIN/AEI/10.13039/501100011033/FEDER, EU, and by the project HERMES, funded by the European Union NextGenerationEU/PRTR via INCIBE.

References

AB87. Noga Alon and Ravi B. Boppana. The monotone circuit complexity of Boolean functions. *Combinatorica*, 7(1):1–22, 1987.

ABI$^+$23. Benny Applebaum, Amos Beimel, Yuval Ishai, Eyal Kushilevitz, Tianren Liu, and Vinod Vaikuntanathan. Succinct computational secret sharing. In *STOC 2023*, pages 1553–1566. ACM, 2023.

AKS83. M. Ajtai, J. Komlós, and E. Szemerédi. An $O(n \log n)$ sorting network. In *Proc. of the 15th ACM Symp. on the Theory of Computing*, pages 1–9, 1983.

BCC$^+$21. Lorenz Breidenbach, Christian Cachin, Benedict Chan, Alex Coventry, Steve Ellis, Ari Juels, Farinaz Koushanfar, Andrew Miller, Brendan Magauran, Daniel Moroz, and et al. Chainlink 2.0: Next steps in the evolution of decentralized oracle networks. Technical report, Chainlink Labs, 2021.

Bei11. Amos Beimel. Secret-sharing schemes: A survey. In Yeow Meng Chee, Zhenbo Guo, San Ling, Fengjing Shao, Yuansheng Tang, Huaxiong Wang, and Chaoping Xing, editors, *Coding and Cryptology – Third International Workshop, IWCC 2011*, volume 6639 of *LNCS*, pages 11–46. Springer-Verlag, 2011.

BF20. Amos Beimel and Oriol Farràs. The share size of secret-sharing schemes for almost all access structures and graphs. *IACR Cryptol. ePrint Arch.*, 2020:664, 2020.

BHS23. Fabrice Benhamouda, Shai Halevi, and Lev Stambler. Weighted secret sharing from wiretap channels. In *4th Conference on Information-Theoretic Cryptography, ITC 2023*, volume 267 of *LIPIcs*, pages 8:1–8:19, 2023.

BL88. Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In Shaffi Goldwasser, editor, *CRYPTO '88*, volume 403 of *LNCS*, pages 27–35. Springer-Verlag, 1988.

Bla79. George Robert Blakley. Safeguarding cryptographic keys. In *Proc. of the 1979 AFIPS National Computer Conference*, volume 48 of *AFIPS Conference proceedings*, pages 313–317. AFIPS Press, 1979.

BTW08. Amos Beimel, Tamir Tassa, and Enav Weinreb. Characterizing ideal weighted threshold secret sharing. *SIAM J. Discrete Math.*, 22(1):360–397, 2008.

BW06. Amos Beimel and E. Weinreb. Monotone circuits for monotone weighted threshold functions. *Inform. Process. Lett.*, 97(1):12–18, 2006. Conference version: *Proc. of 20th Annu. IEEE Conf. on Computational Complexity*, pages 67-75, 2005.

CM17. Jing Chen and Silvio Micali. Algorand, 2017.

COS17. Xi Chen, Igor C. Oliveira, and Rocco A. Servedio. Addition is exponentially harder than counting for shallow monotone circuits. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing,* STOC 2017, page 1232–1245, New York, NY, USA, 2017. Association for Computing Machinery.

FG24. Oriol Farràs and Miquel Guiot. Reducing the share size of weighted threshold secret sharing schemes via chow parameters approximation. In *TCC*, 2024.

FMFP12. O. Farràs, J. Martí-Farré, and C. Padró. Ideal multipartite secret sharing schemes. *Journal of cryptology*, 25(3):434–463, 2012.

FP12. Oriol Farràs and Carles Padró. Ideal hierarchical secret sharing schemes. *IEEE Transactions on Information Theory*, 58(5):3273–3286, 2012.

GJM$^+$23. Sanjam Garg, Abhishek Jain, Pratyay Mukherjee, Rohit Sinha, Mingyuan Wang, and Yinuo Zhang. Cryptography with weights: Mpc, encryption and signatures. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023*, pages 295–327, Cham, 2023. Springer Nature Switzerland.

GK93. M. Goldmann and M. Karpinski. Simulating threshold circuits by majority circuits. In *Proc. of the 15th ACM Symp. on the Theory of Computing*, pages 551–560, 1993.

Hås94. Johan Håstad. On the size of weights for threshold gates. *SIAM J. on Discrete Mathematics*, 7(3):484–492, 1994.

Hof92. T. Hofmeister. The power of negative thinking in constructing threshold circuits for addition. In *[1992] Proceedings of the Seventh Annual Structure in Complexity Theory Conference*, pages 20–26, 1992.

Kra94. H. Krawczyk. Secret sharing made short. In *CRYPTO '93*, volume 773 of *LNCS*, pages 136–146. Springer-Verlag, 1994.

KRDO17. Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, pages 357–388, Cham, 2017. Springer International Publishing.

Lab22. Aptos Labs. The aptos blockchain: Safe, scalable, and upgradeable web3 infrastructure, 2022.

Mur71. Saburo Muroga. *Threshold Logic and Its Applications.* Wiley-Interscience, 1971.

PS00. Carles Padró and Germán Sáez. Secret sharing schemes with bipartite access structure. *IEEE Transactions on Information Theory*, 46(7):2596–2604, 2000.

Raz87. Alexander A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41:333–338, 1987.

RYS$^+$20. Team Rocket, Maofan Yin, Kevin Sekniqi, Robbert van Renesse, and Emin Gün Sirer. Scalable and probabilistic leaderless bft consensus through metastability, 2020.

Sha79. Adi Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, 1979.

TF24. Andrei Tonkikh and Luciano Freitas. Swiper: a new paradigm for efficient weighted distributed protocols. In *Proceedings of the 43rd ACM Symposium on Principles of Distributed Computing*, PODC '24, page 283–294. Association for Computing Machinery, 2024.

Val84. L. G. Valiant. Short monotone formulae for the majority function. *J. of Algorithms*, 5(3):363–366, 1984.

Woo14. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 2014.

Yak17. Anatoly Yakovenko. Solana: A new architecture for a high performance blockchain, 2017.

Yao89. A. C. Yao. Unpublished manuscript, 1989. Presented at Oberwolfach and DIMACS workshops.

A Illustrative Examples of the Circuits of Section 3

In this section, we provide concrete instantiations of the circuits of Section 3 for $f(\boldsymbol{x}) = \mathrm{sign}(4x_1 + 2x_2 + 2x_3 + 2x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} - 8)$.

A.1 Monotone Majority Boolean Circuit of Proposition 3.5

Fig. 4 illustrates the instantiation of the circuit presented in Proposition 3.5 for $f(\boldsymbol{x}) = \mathrm{sign}(4x_1 + 2x_2 + 2x_3 + 2x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} - 8)$.

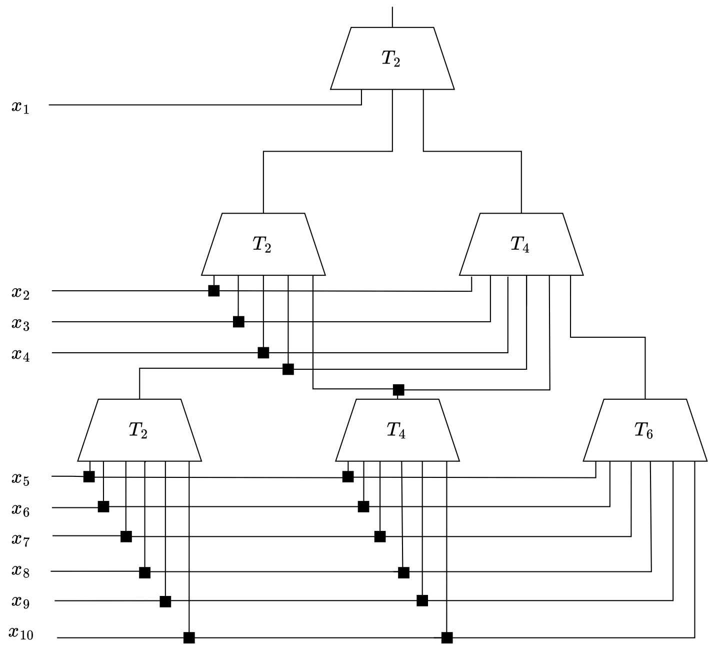

Fig. 4: Monotone majority Boolean circuit realizing the WTAS given by $f(\boldsymbol{x}) = \mathrm{sign}(4x_1 + 2x_2 + 2x_3 + 2x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} - 8)$.

A.2 Monotone Majority Boolean Circuit of Proposition 3.7

Fig. 3 illustrates the optimization presented in Section 3.3 for the circuit of Fig. 2.

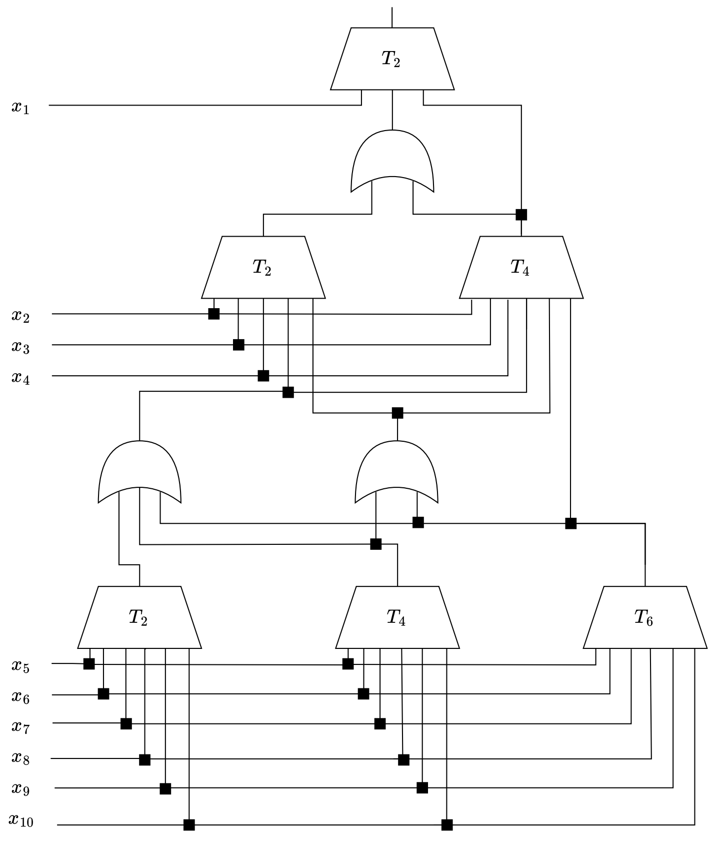

Fig. 5: Optimized version of the circuit realizing the WTAS given by $f(\boldsymbol{x}) = \text{sign}(4x_1 + 2x_2 + 2x_3 + 2x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} - 8)$.

B Weighted Threshold Schemes from Monotone Formulas

The depth of our polynomial-size monotone majority Boolean circuit is $O(\log \sigma)$. Hence, in the worst case, the depth is superlinear in the number of parties. However, since it corresponds to an improvement and simplification of the proposal of Beimel and Weinreb [BW06], we can shorten its depth by performing the same balancing and dynamic techniques as in their work.

In more detail, the circuit depth can be reduced using a standard balancing technique. This involves computing additions across multiple levels in parallel, using several candidate values for the carry, and determining the correct output with a small selector circuit. Although this process increases the circuit's size, it is subsequently reduced to a polynomial size by properly wiring it using dynamic programming techniques. As a consequence, any monotone WTF over n variables can be computed by a majority Boolean circuit with $\mathrm{poly}(n)$ size and $O(\log\log\sigma)$ depth. For the full details, we refer to the work of Beimel and Weinreb [BW06].

Next, we apply a black-box transformation that converts circuits into formulas to this circuit. This yields a monotone Boolean formula of quasipolynomial size. We then employ the compiler by Benaloh and Leichter [BL88], which constructs information-theoretic secret sharing schemes from monotone formulas. These steps collectively lead to the fact that any WTAS over n parties and threshold σ admits an information-theoretic secret sharing scheme with share size $n^{O(\log\log\sigma)}$. This bound coincides with the one in [BW06].

Remark B.1. In the worst case, i.e. when $\log\sigma = O(n\log n)$, the polynomial-size logarithmic-depth monotone majority Boolean circuit we obtain is equivalent to the one presented in the work of Chen, Oliveira and Servedio [COS17]. In this sense, although the depth of the resulting circuit is smaller than that of Section 3.2, they show that its size is larger, i.e. of the order of $O(n^{18}\log^{6}(n))$. Therefore, applying Yao's compiler to it would lead to a computational secret sharing scheme with larger public information than that of Section 3.2.

Moreover, in their work, Chen, Oliveira and Servedio [COS17] also proved the result stated in Theorem 2.3, which guarantees that not all monotone WTFs can be computed by constant depth polynomial-size monotone majority Boolean circuits. Essentially, this result is a step towards proving that no black-box transformation from monotone circuits to monotone formulas will lead to a polynomial-size monotone formula for all WTF s. Therefore, it suggests a limitation in the technique used in this work and that of Beimel and Weinreb [BW06] to construct information-theoretic schemes for WTASs.

C Recursive Secret Sharing Schemes for Weighted Threshold Access Structures

In this section, we provide a recursive information-theoretic scheme for WTASs that outperforms Shamir's virtualization technique for some weight distributions. To do so, we first provide a motivating example where this alternative approach is more appropriate. Next, we formally prove the correctness and privacy of the scheme by providing a recursive monotone formula for WTASs.

C.1 A Motivating Example

Applying Shamir's virtualization technique in the weighted setting leads to a secret sharing scheme in which the share size grows linearly with the weights of the parties. This can be problematic for two reasons: First, as shown in Theorem 2.5, the size of the weights can be exponential in the number of parties. Second, this approach penalizes the parties with the largest weights, as they receive the largest shares.

For example, consider the WTAS in which the threshold is set to $\sigma = 20$ and there are a total of $n = 24$ parties: $n_1 = 3$ of them with weight $w_1 = 10$ and the remaining $n_2 = 21$ with weight $w_2 = 1$.

Shamir's Virtualization Technique. If we apply Shamir's virtualization technique to this access structure, we end up with a total share size of

$$(w_1 \cdot n_1 + w_2 \cdot n_2) \log(w_1 \cdot n_1 + w_2 \cdot n_2) = (10 \cdot 3 + 1 \cdot 21) \log(10 \cdot 3 + 1 \cdot 21) = 51 \log(51),$$

where each of the three parties with the largest weight gets shares ten times larger than the rest of the parties.

However, a more fine-grained analysis of the access structure shows that there exists an alternative scheme that leads to smaller total share size and where the parties with the largest weights get simply one share.

Alternative Construction. At a high level, the idea behind our proposal is to exploit the following two facts: First, notice that a group of 10 parties with weight $w_2 = 1$ is equivalent to a party of weight $w_1 = 10$. Second, notice that any two parties with weight $w_2 = 10$ reach the threshold $\sigma = 20$. With this in mind, we can simply perform a standard 2-threshold secret sharing among the parties with the largest weights, produce two extra shares for virtual parties, and share them among the real parties with the lowest weight in such a way that the 10-to-1 relation holds. The actual procedure follows.

Since the threshold is $\sigma = 20$ and there are three parties with the largest weight $w_1 = 10$, we compute

$$\sigma_1 = \frac{\sigma}{w_1} = \frac{20}{10} = 2,$$

apply a σ_1-threshold secret sharing scheme to these $n_1 = 3$ parties and produce $\sigma_1 = 2$ additional shares sh_1 and sh_2. Next, we compute

$$\sigma_2 = \frac{w_1}{w_2} = \frac{10}{1} = 10 \text{ and } \sigma_3 = 2 \cdot \frac{w_1}{w_2} = 2 \cdot \frac{10}{1} = 20,$$

and secret share sh_1 and sh_2 among the remaining $n_2 = 21$ parties with weight $w_2 = 1$ using a σ_2-threshold and a σ_3-threshold secret sharing scheme respectively. It is straightforward to check that this construction realizes the same access structure and that it leads to a total share size of

$$n_1 \log(n_1) + 2n_2 \log(n_2) = 3 \log(3) + 2 \cdot 21 \log(21) = 3 \log(3) + 42 \log(21),$$

where each of the three parties with the largest weight gets simply one share.

C.2 On the Generalization of the Recursive Construction

In the example above, the two observations that formed the basis of our proposal required the threshold and all the weights to be multiples of each other. At first glance, this may seem like a significant limitation, as most WTAS do not exhibit such a relationship between the weights and the threshold. However, recall that the reduction in Theorem 3.2 enables the transformation of any WTAS into another in which the threshold and weights are decreasing powers of two. Therefore, to overcome this limitation we simply need to apply this reduction as a first step.

With this in mind, we now formally state and prove the extension of this recursive scheme to the general case.

The secret: An element $s \in \{0,1\}^{\lceil \log n \rceil + 1}$.

The access structure: A WTAS over n parties given by $f(\boldsymbol{x}) = \mathrm{WTF}(\boldsymbol{w}, \sigma)(\boldsymbol{x})$, where the threshold σ and the coordinates of $\boldsymbol{w}$ are powers of two, and participants are partitioned into $\Pi = (P_1, \ldots, P_\ell)$ according to their weight.

The scheme $\Sigma'(s, \boldsymbol{w}, \sigma, \Pi)$:

1. Let $q = 2^{\lceil \log n \rceil + 1}$, $\sigma_1 = \frac{\sigma}{w_1}$ be the threshold scaled down by w_1, let $a_1 = |P_1|$, and let $n_1 = \lfloor \frac{1}{w_1} \sum_{i \in P \setminus P_1} w_i \rfloor$ Compute $m_1 = \min(\sigma_1, n_1)$.
2. Let P_1^{aux} be m_1 auxiliary parties of weight w_1 and let $\boldsymbol{r} \in \mathbb{F}_q^{\sigma_1 - 1}$ be a uniformly random string. Secret share s among $P_1 \cup P_1^{\mathrm{aux}}$ using Shamir's secret sharing scheme with threshold σ_1 and randomness $\boldsymbol{r}$, i.e.
$$\mathsf{sh}_1, \ldots, \mathsf{sh}_{a_1+m_1} \leftarrow \mathsf{Shamir}(s, \boldsymbol{r}, \sigma_1).$$
3. For $1 \leq i \leq a_1$, send share sh_i to participant i of P_1.
4. If $\ell \neq 1$, i.e. if there is more than one weight, for $i \in [m_1]$ do a recursive call of the scheme Σ' with secret sh_{a_1+i}, threshold $i \cdot w_1$, weights $\boldsymbol{w}' = \boldsymbol{w}_{P \setminus P_1}$, and parties $\Pi' = \Pi \setminus P_1$, i.e.
$$\text{for } i \in [m_1] \text{ execute } \Sigma'(\mathsf{sh}_{a_1+i}, \boldsymbol{w}', i \cdot w_1, \Pi').$$

Fig. 6: Recursive secret sharing scheme for WTASs whose threshold and weights are decreasing powers of two.

First, we prove the correctness and privacy of the scheme for the case in which the weights and the threshold are decreasing powers of two.

Proposition C.1. *Let Γ be a WTAS over n parties given by $f(\boldsymbol{x}) = \mathrm{WTF}(\boldsymbol{w}, \sigma)(\boldsymbol{x})$ where $\sigma, \boldsymbol{w}$ are decreasing powers of two and let $\Pi = \{P_1, \ldots, P_\ell\}$ be a partition of the parties according to their weights. For any $s \in \{0,1\}^{\lceil \log n \rceil + 1}$ the protocol*

The secret: An element $s \in \{0,1\}^{\lceil \log n \rceil + 1}$.

The access structure: A WTAS over n parties given by $f(\boldsymbol{x}) = \mathrm{WTF}(\boldsymbol{w}, \sigma)(\boldsymbol{x})$.

The scheme $\Sigma(s, \boldsymbol{w}, \sigma)$:

1. Apply the $\mathsf{Reduction}$ of Theorem 3.2 to $f(\boldsymbol{x})$ to obtain a monotone WTF $g(\boldsymbol{y}) = \mathrm{WTF}(\boldsymbol{w}', \sigma')(\boldsymbol{y})$, where the threshold σ' and the weights $\boldsymbol{w}'$ are decreasing powers of two, i.e.
$$g(\boldsymbol{y}) \leftarrow \mathsf{Reduction}(f(\boldsymbol{x})).$$
2. Let Π be a partition of the parties in g according to their weights $\boldsymbol{w}'$, and let $S_i \subseteq [O(n \log(n))]$ be the set such that $w_i = \sum_{j \in S_i} 2^j$. Secret share s among the parties in g using the scheme Σ' of Fig. 6, i.e.
$$\mathsf{sh}_1^1, ..., \mathsf{sh}_n^{S_n} \leftarrow \Sigma'(s, \boldsymbol{w}', \sigma, \Pi).$$

Share distribution: The share of the i-th party is $(\mathsf{sh}_i^j)_{j \in S_i}$.

Fig. 7: Recursive secret sharing scheme for any WTASs.

$\Sigma'(s, \boldsymbol{w}, \sigma, \Pi)$ presented in Fig. 6 is a secret sharing scheme realizing Γ whose share size is $n^{O(\log(\sigma))}$.

Proof. First, notice that to prove the correctness and privacy of the scheme it suffices to give a description of f in terms of a monotone Boolean formula with threshold gates that realizes precisely the scheme depicted in Fig. 6, since then we can simply use the transformation of monotone boolean formulas into secret sharing schemes by Benaloh and Leichter [BL88].

With this in mind, since several participants may have the same weight, we use the partition $\Pi = (P_1, \dots, P_\ell)$ where the participants in P_i have the same weight $w_i = 2^i$, and $\ell = \lceil \log \sigma \rceil$. In this case, the access structure is ℓ-partite [FMFP12] and, to make it evident, it is convenient to describe the function as $F(\boldsymbol{w}, \sigma, \Pi) = WTF(\boldsymbol{w}, \sigma)$.

Taking advantage of this description, we can describe the function recursively as follows. The authorized subsets of the WTF can be classified according to the number of participants they have in each part. For instance, the authorized subsets are those that do not have any participant in P_1 but that satisfy that the sum of their weights is at least σ, those that have one participant in P_1 and that the sum of weights of the remaining participants is at least $\sigma - w_1$; and so on. In each one of the cases, we are considering the AND of two functions: The ℓ-threshold access structure on the set P_1 and the WTAS on $\Pi \backslash P_1$ and threshold $\sigma - kw_1$ for some $0 \leq k \leq |P_1|$ (which is a $(P_2, \dots, P_\ell)$-partite function).

Therefore, we can define $F(\boldsymbol{w}, \sigma, \Pi)$ recursively in ℓ as follows. If $\ell = 1$, then $\Pi = (P_1)$ and
$$F(\boldsymbol{w}, \sigma, \Pi)(\boldsymbol{x}) = \mathrm{Th}\left(\frac{\sigma}{w_1}\right)(\boldsymbol{x}).$$

If $\ell > 1$, then

$$F(\boldsymbol{w}, \sigma, \Pi)(\boldsymbol{x}) = \bigvee_{j=0}^{\frac{\sigma}{w_1}} \mathrm{Th}\left(\frac{\sigma}{w_1} - j\right)(\boldsymbol{x}') \wedge F(\boldsymbol{w}', jw_1, \Pi')(\boldsymbol{x}'') \tag{1}$$

where $\boldsymbol{w}' = \boldsymbol{w}_{P \setminus P_1}$, $\Pi' = \Pi_{[2,\ell]}$, $\boldsymbol{x}' = \boldsymbol{x}_{P_1}$, and $\boldsymbol{x}'' = \boldsymbol{x}_{P \setminus P_1}$.

Now, we present an alternative formulation of this recursive function to prove that indeed the scheme in Fig. 6 realizes this access structure. To simplify the notation, let $f_j = F(\boldsymbol{w}', jw_1, \Pi')$ for $0 \leq j < \frac{\sigma}{w_1}$. Notice that $f_j \leq f_{j+1}$ for every $0 \leq j < \frac{\sigma}{w_1}$. Therefore, the j-threshold function with inputs $f_1, \ldots, f_{\frac{\sigma}{w_1}}$ is equal to f_j. That is,

$$\mathrm{Th}(j)(f_1(\boldsymbol{x}''), \ldots, f_{\frac{\sigma}{w_1}}(\boldsymbol{x}'')) = f_j(\boldsymbol{x}'').$$

In some cases, for high j, it is possible that $f_j(\boldsymbol{x}'') = 0$ because the weight of the remaining participants is not big enough, i.e., when $|\boldsymbol{w}'| < jw_1$. Therefore, the index j only needs to run up to $m_1 = \min(\sigma_1, n_1)$, where $\sigma_1 = \frac{\sigma}{w_1}$ and $n_1 = \frac{|\boldsymbol{w}'|}{w_1}$. With this observations in mind, the expression (1) can be rewritten as

$$\begin{aligned} F(\boldsymbol{w}, \sigma, \Pi)(\boldsymbol{x}) &= \bigvee_{j=0}^{m_1} \mathrm{Th}\left(\frac{\sigma}{w_1} - j\right)(\boldsymbol{x}') \wedge f_j(\boldsymbol{x}'') \\ &= \bigvee_{j=0}^{m_1} \mathrm{Th}\left(\frac{\sigma}{w_1} - j\right)(\boldsymbol{x}') \wedge \mathrm{Th}(j)(f_1(\boldsymbol{x}''), \ldots, f_{m_1}(\boldsymbol{x}'')) \\ &= \mathrm{Th}\left(\frac{\sigma}{w_1}\right)(\boldsymbol{x}' || (f_1(\boldsymbol{x}''), \ldots, f_{m_1}(\boldsymbol{x}''))), \end{aligned}$$

where in the last equality we use the property that for any two disjoint sets Q and Q' we have that

$$\bigcup_{i=0}^{k} \left\{ A \cup B \,:\, A \in \binom{Q}{i}, B \in \binom{Q'}{k-i} \right\} = \binom{Q \cup Q'}{k}.$$

The advantage of this new description of $F(\boldsymbol{w}, \sigma, \Pi)$ is that it is just the threshold function of certain inputs. Applying the recursion argument, we get that the WTF $f(\boldsymbol{x}) = WTF(\boldsymbol{w}, \sigma)(\boldsymbol{x})$ can be described as a monotone Boolean formula with threshold gates. Following the transformation of monotone boolean formulas into secret sharing schemes by Benaloh and Leichter [BL88], we obtain the scheme described in Fig. 6.

It only remains to prove the bound on the share size, which corresponds to the number of leaves appearing in the tree description of the recursive formula. From this description, it is clear that the formula has at most $\ell = \lceil \log \sigma \rceil$ levels and that the factor of expansion is at most $2n$. Therefore, each level contains at most $(2n)^\ell$ leaves, which implies that there are at most $\ell(2n)^\ell = n^{O(\log \sigma)}$ leaves in total. □

Now, we move to prove the correctness and privacy of the scheme for the general case.

Theorem C.2. *Let Γ be a WTAS over n parties given by $f(\boldsymbol{x}) = \mathrm{WTF}(\boldsymbol{w}, \sigma)(\boldsymbol{x})$. For any $s \in \{0,1\}^{\lceil \log n \rceil + 1}$ the protocol $\Sigma(s, \boldsymbol{w}, \sigma)$ presented in Fig. 7 is a secret sharing scheme realizing Γ whose share size is $n^{O(\log \sigma)}$.*

Proof. The correctness and privacy of the scheme follow from combining Theorem 3.2 and Proposition C.1. □